#6
50
5/29/01

JP01/408

PCT/JP01/00408

日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT

28.01.01

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日
Date of Application: ２０００年 ７月１４日

出 願 番 号
Application Number: 特願２０００－２１３９８８

出 願 人
Applicant (s): 株式会社ナムコ

２００１年 ２月２３日

特 許 庁 長 官
Commissioner,
Patent Office 及 川 耕 造

出証番号 出証特２００１－３００９６８８

【書類名】　特許願

【整理番号】　NM-0132401

【提出日】　平成12年 7月14日

【あて先】　特許庁長官殿

【国際特許分類】　A63F　9/22

【発明者】

【住所又は居所】　東京都大田区多摩川２丁目８番５号　株式会社ナムコ内

【氏名】　橘高　繁

【特許出願人】

【識別番号】　000134855

【氏名又は名称】　株式会社ナムコ

【代理人】

【識別番号】　100090387

【弁理士】

【氏名又は名称】　布施　行夫

【電話番号】　03-5397-0891

【選任した代理人】

【識別番号】　100090479

【弁理士】

【氏名又は名称】　井上　一

【電話番号】　03-5397-0891

【選任した代理人】

【識別番号】　100090398

【弁理士】

【氏名又は名称】　大渕　美千栄

【電話番号】　03-5397-0891

【先の出願に基づく優先権主張】

【出願番号】　特願2000- 20464

【出願日】　平成12年 1月28日

【手数料の表示】

【予納台帳番号】　039479

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9814051

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　ゲームシステム及び情報記憶媒体

【特許請求の範囲】

【請求項１】　画像生成を行うゲームシステムであって、

元画像の各画素の奥行き値を、インデックスカラー・テクスチャマッピング用のルックアップテーブルのインデックス番号として設定する手段と、

元画像の各画素の奥行き値がインデックス番号として設定された前記ルックアップテーブルを用いて、仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、元画像の各画素の奥行き値に応じた値に各画素のα値を設定する手段と、

元画像と該元画像に対応するぼかし画像とを、各画素に設定されるα値に基づいて合成する手段と、

を含むことを特徴とするゲームシステム。

【請求項２】　請求項１において、

元画像の各画素の奥行き値を、前記奥行き値の最上位ビットよりも下位のビットＩ～ビットＪにより構成される第２の奥行き値に変換し、前記第２の奥行き値を、インデックスカラー・テクスチャマッピング用の前記ルックアップテーブルのインデックス番号として設定することを特徴とするゲームシステム。

【請求項３】　請求項２において、

前記奥行き値のビットＩ～ビットＪ以外のビットの値に応じて、前記第２の奥行き値を所与の値にクランプすることを特徴とするゲームシステム。

【請求項４】　請求項２又は３において、

前記奥行き値を、インデックスカラー・テクスチャマッピング用のルックアップテーブルのインデックス番号として設定し、前記ルックアップテーブルを用いて仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、前記奥行き値を前記第２の奥行き値に変換することを特徴とするゲームシステム。

【請求項５】　請求項２乃至４のいずれかにおいて、

前記奥行き値のビットＭ～ビットＮを、インデックスカラー・テクスチャマッピング用の第１のルックアップテーブルのインデックス番号として設定し、前記第１のルックアップテーブルを用いて仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、前記奥行き値を第３の奥行き値に変換し、

前記奥行き値のビットＫ～ビットＬ（Ｋ≧Ｉ≧Ｌ＞Ｍ≧Ｊ≧Ｎ）を、インデックスカラー・テクスチャマッピング用の第２のルックアップテーブルのインデックス番号として設定し、前記第２のルックアップテーブルを用いて仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、前記奥行き値を第４の奥行き値に変換し、

前記第３、第４の奥行き値に基づいて前記第２の奥行き値を求めることを特徴とするゲームシステム。

【請求項６】　請求項１乃至５のいずれかにおいて、

元画像をテクスチャとして設定し、該テクスチャをテクセル補間方式で仮想オブジェクトにマッピングする際に仮想オブジェクトのテクスチャ座標をシフトさせ、元画像のぼかし画像を生成することを特徴とするゲームシステム。

【請求項７】　請求項１乃至６のいずれかにおいて、

前記仮想オブジェクトが、表示画面サイズのポリゴンであることを特徴とするゲームシステム。

【請求項８】　請求項１乃至６のいずれかにおいて、

前記仮想オブジェクトが、表示画面を分割したブロックのサイズのポリゴンであることを特徴とするゲームシステム。

【請求項９】　コンピュータが使用可能な情報記憶媒体であって、

元画像の各画素の奥行き値を、インデックスカラー・テクスチャマッピング用のルックアップテーブルのインデックス番号として設定する手段と、

元画像の各画素の奥行き値がインデックス番号として設定された前記ルックアップテーブルを用いて、仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、元画像の各画素の奥行き値に応じた値に各画素のα値を設定する手段と、

元画像と該元画像に対応するぼかし画像とを、各画素に設定されるα値に基づ

いて合成する手段と、

を実行するためのプログラムを含むことを特徴とする情報記憶媒体。

【請求項１０】　請求項９において、

元画像の各画素の奥行き値を、前記奥行き値の最上位ビットよりも下位のビットＩ～ビットＪにより構成される第２の奥行き値に変換し、前記第２の奥行き値を、インデックスカラー・テクスチャマッピング用の前記ルックアップテーブルのインデックス番号として設定することを特徴とする情報記憶媒体。

【請求項１１】　請求項１０において、

前記奥行き値のビットＩ～ビットＪ以外のビットの値に応じて、前記第２の奥行き値を所与の値にクランプすることを特徴とする情報記憶媒体。

【請求項１２】　請求項１０又は１１において、

前記奥行き値を、インデックスカラー・テクスチャマッピング用のルックアップテーブルのインデックス番号として設定し、前記ルックアップテーブルを用いて仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、前記奥行き値を前記第２の奥行き値に変換することを特徴とする情報記憶媒体。

【請求項１３】　請求項１０乃至１２のいずれかにおいて、

前記奥行き値のビットＭ～ビットＮを、インデックスカラー・テクスチャマッピング用の第１のルックアップテーブルのインデックス番号として設定し、前記第１のルックアップテーブルを用いて仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、前記奥行き値を第３の奥行き値に変換し、

前記奥行き値のビットＫ～ビットＬ（Ｋ≧Ｉ≧Ｌ＞Ｍ≧Ｊ≧Ｎ）を、インデックスカラー・テクスチャマッピング用の第２のルックアップテーブルのインデックス番号として設定し、前記第２のルックアップテーブルを用いて仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、前記奥行き値を第４の奥行き値に変換し、

前記第３、第４の奥行き値に基づいて前記第２の奥行き値を求めることを特徴とする情報記憶媒体。

【請求項１４】　請求項９乃至１３のいずれかにおいて、

元画像をテクスチャとして設定し、該テクスチャをテクセル補間方式で仮想オブジェクトにマッピングする際に仮想オブジェクトのテクスチャ座標をシフトさせ、元画像のぼかし画像を生成することを特徴とする情報記憶媒体。

【請求項１５】　請求項９乃至１４のいずれかにおいて、

前記仮想オブジェクトが、表示画面サイズのポリゴンであることを特徴とする情報記憶媒体。

【請求項１６】　請求項９乃至１４のいずれかにおいて、

前記仮想オブジェクトが、表示画面を分割したブロックのサイズのポリゴンであることを特徴とする情報記憶媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、ゲームシステム及び情報記憶媒体に関する。

【０００２】

【背景技術及び発明が解決しようとする課題】

従来より、仮想的な３次元空間であるオブジェクト空間内の所与の視点から見える画像を生成するゲームシステムが知られており、いわゆる仮想現実を体験できるものとして人気が高い。レーシングゲームを楽しむことができるゲームシステムを例にとれば、プレーヤは、車（オブジェクト）を操作してオブジェクト空間内で走行させ、他のプレーヤやコンピュータが操作する車と競争することで３次元ゲームを楽しむ。

【０００３】

さて、従来のゲームシステムにより生成される画像は、人間の視界画像のように視点からの距離に応じてフォーカシングされた画像ではなかった。このため、画像内の全ての被写体にピントが合っているかのような表現になっていた。

【０００４】

しかしながら、至近距離から遠距離までの全ての被写体にピントが合っている画像は、日常生活では見ることができない画像であるため、見た目に不自然さがあった。

【０００５】

よりリアリティを追求するためには、視点とオブジェクトとの距離や視線方向などに応じてピントの度合いが調節された画像を生成することが望ましい。しかしながら、ゲーム空間内の個々のオブジェクトについて視点との距離等を計算し、各オブジェクト毎にぼやけ具合を演算することで、ぼやけた画像を生成すると、処理負荷が過大になる。

【０００６】

リアルタイムに変化する視点に対応した画像を、制約されたハードウェア資源を用いて生成する必要があるゲームシステムにおいては、如何にして少ない処理負担で、現実世界の視界画像のようにフォーカシングされた画像を生成するかが重要な課題となる。

【０００７】

本発明は、以上のような課題に鑑みてなされたものであり、その目的とするところは、よりリアルな画像を少ない処理負担で生成できるゲームシステム及び情報記憶媒体を提供することにある。より具体的には、現実世界の視界画像のようにフォーカシングされた画像を、少ない処理負担で生成できるゲームシステム及び情報記憶媒体を提供することにある。

【０００８】

【課題を解決するための手段】

上記課題を解決するために、本発明は、画像生成を行うゲームシステムであって、元画像の各画素の奥行き値を、インデックスカラー・テクスチャマッピング用のルックアップテーブルのインデックス番号として設定する手段と、元画像の各画素の奥行き値がインデックス番号として設定された前記ルックアップテーブルを用いて、仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、元画像の各画素の奥行き値に応じた値に各画素のα値を設定する手段と、元画像と該元画像に対応するぼかし画像とを、各画素に設定されるα値に基づいて合成する手段とを含むことを特徴とする。また本発明に係る情報記憶媒体は、コンピュータにより使用可能な情報記憶媒体であって、上記手段を実行するためのプログラムを含むことを特徴とする。また本発明に係るプログラムは、

コンピュータにより使用可能なプログラム（搬送波に具現化されるプログラムを含む）であって、上記手段を実行するための処理ルーチンを含むことを特徴とする。

【０００９】

本発明によれば、元画像の各画素の奥行き値がインデックス番号として設定されたルックアップテーブルを用いて、仮想オブジェクトに対してインデックスカラー・テクスチャマッピングが行われ、元画像の各画素の奥行き値に応じた値に各画素のα値が設定される。このように本発明によれば、ゲームシステム（画像生成システム）が元々有しているインデックスカラー・テクスチャマッピングの機能を有効利用して、奥行き値をα値に変換できる。従って、奥行き値の変換処理を、例えば新たなハードウェアを追加することなく、高速に実行できるようになり、全表示画面分の奥行き値の変換も容易となる。

【００１０】

そして本発明によれば、元画像の各画素の奥行き値に応じた値に設定されたα値に基づいて、元画像とぼかし画像とが合成される。従って、奥行き値に応じて、ぼかし画像の合成比率等を変化させることが可能になり、被写界深度などの表現が可能になる。

【００１１】

なお、α（アルファ）値は、各画素に関連づけられて記憶される情報であり、例えば色情報以外の情報である。また奥行き値は、視点（仮想カメラ）から近いほど大きい値にしてもよいし、視点から遠いほど大きい値にしてもよい。また、元画像の合成対象となるぼかし画像の合成手法としては、種々の手法を採用できる。またα値を用いた合成処理はαブレンディングに限定されない。

【００１２】

また本発明に係るゲームシステム、情報記憶媒体及びプログラムは、元画像の各画素の奥行き値を、前記奥行き値の最上位ビットよりも下位のビットＩ～ビットＪにより構成される第２の奥行き値に変換し、前記第２の奥行き値を、インデックスカラー・テクスチャマッピング用の前記ルックアップテーブルのインデックス番号として設定することを特徴とする。

【００１３】

このようにすれば、α値の実効的な区分け数（奥行き値のしきい値の実効的な段階数）を増やすこと可能となり、例えば、仮想カメラの焦点（注視点）付近にあるオブジェクトのぼけ具合を、多段階のしきい値で精度良く制御できるようになる。従って、生成される画像の品質を高めることができる。

【００１４】

また本発明に係るゲームシステム、情報記憶媒体及びプログラムは、前記奥行き値のビットＩ～ビットＪ以外のビットの値に応じて、前記第２の奥行き値を所与の値にクランプすることを特徴とする。

【００１５】

このようにすれば、奥行き値のビットＩ～ビットＪ以外のビットに１が立った場合等においても、矛盾の無い画像を生成できる。なお、所与の値としては、第２の奥行き値の最大値や最小値、或いは第２の奥行き値の上位のビット列を１にした値など、種々の値を考えることができる。

【００１６】

また本発明に係るゲームシステム、情報記憶媒体及びプログラムは、前記奥行き値を、インデックスカラー・テクスチャマッピング用のルックアップテーブルのインデックス番号として設定し、前記ルックアップテーブルを用いて仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、前記奥行き値を前記第２の奥行き値に変換することを特徴とする。

【００１７】

このようにすれば、ルックアップテーブルの変換特性を変更するだけという少ない処理負担で、第２の奥行き値を所与の値にクランプするなどの多様な変換処理を実現できるようになる。

【００１８】

また本発明に係るゲームシステム、情報記憶媒体及びプログラムは、前記奥行き値のビットＭ～ビットＮを、インデックスカラー・テクスチャマッピング用の第１のルックアップテーブルのインデックス番号として設定し、前記第１のルックアップテーブルを用いて仮想オブジェクトに対してインデックスカラー・テク

スチャマッピングを行い、前記奥行き値を第３の奥行き値に変換し、前記奥行き値のビットＫ～ビットＬ（Ｋ≧Ｉ≧Ｌ＞Ｍ≧Ｊ≧Ｎ）を、インデックスカラー・テクスチャマッピング用の第２のルックアップテーブルのインデックス番号として設定し、前記第２のルックアップテーブルを用いて仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、前記奥行き値を第４の奥行き値に変換し、前記第３、第４の奥行き値に基づいて前記第２の奥行き値を求めることを特徴とする。

【００１９】

このようにすれば、奥行き値の所定範囲のビット列（例えば０～７、８～１５、１６～２３、２４～３１ビット）しか取り出せないような制限がある場合においても、奥行き値の任意のビットＩ～Ｊから構成される第２の奥行き値を得ることができる。これにより、α値を区分けする奥行き値のしきい値の実効的な段階数を増やすこと可能となり、生成される画像の品質を高めることができる。

【００２０】

また本発明に係るゲームシステム、情報記憶媒体及びプログラムは、元画像をテクスチャとして設定し、該テクスチャをテクセル補間方式で仮想オブジェクトにマッピングする際に仮想オブジェクトのテクスチャ座標をシフトさせ、元画像のぼかし画像を生成することを特徴とする。

【００２１】

このようにすれば、テクスチャ座標をシフトさせながらテクセル補間方式で仮想オブジェクトに元画像をマッピングするだけという簡素な処理で、元画像のぼかし画像を生成できるようになる。

【００２２】

なお、テクセル補間方式とは、特には限定はされないが、テクセルの画像情報を補間してピクセルの画像情報を得る方式などであり、例えば、バイリニアフィルタ方式やトライリニアフィルタ方式などを考えることができる。

【００２３】

また、仮想オブジェクトは、ポリゴンなどのプリミティブ面であることが望ましいが、立体的なオブジェクトであってもよい。また、仮想オブジェクトは画面

上に表示しないことが望ましいが、表示するようにしてもよい。

【００２４】

また、テクセル補間方式のテクスチャマッピングでぼかし画像を生成する場合には、テクスチャ座標を１テクセルよりも小さい値だけシフトさせることが望ましい。また、テクスチャ座標を第１のシフト方向へシフトさせてテクセル補間方式でテクスチャマッピングを行った後に、テクスチャ座標を第２のシフト方向にシフトさせてテクセル補間方式でテクスチャマッピングを行ってもよい。或いは、第１のシフト方向へのシフトと第２のシフト方向へのシフトのセットを、複数回繰り返してもよい。

【００２５】

また本発明に係るゲームシステム、情報記憶媒体及びプログラムは、前記仮想オブジェクトが、表示画面サイズのポリゴンであることを特徴とする。

【００２６】

このようにすれば、全表示画面分の元画像の奥行き値を例えば１回（或いは数回）のテクスチャマッピングでα値に変換できるようになる。

【００２７】

また本発明に係るゲームシステム、情報記憶媒体及びプログラムは、前記仮想オブジェクトが、表示画面を分割したブロックのサイズのポリゴンであることを特徴とする。

【００２８】

このようにすれば、仮想オブジェクトを描画する領域の大きさを小さくすることが可能になり、記憶部の使用記憶容量を節約できる。

【００２９】

【発明の実施の形態】

以下、本発明の好適な実施形態について図面を用いて説明する。

【００３０】

１．構成

図１に、本実施形態のゲームシステム（画像生成システム）のブロック図の一例を示す。なお同図において本実施形態は、少なくとも処理部１００を含めばよ

く、それ以外のブロックについては、任意の構成要素とすることができる。

【００３１】

ここで処理部１００は、システム全体の制御、システム内の各ブロックへの命令の指示、ゲーム処理、画像処理、音処理などの各種の処理を行うものであり、その機能は、各種プロセッサ（ＣＰＵ、ＤＳＰ等）、或いはＡＳＩＣ（ゲートアレイ等）などのハードウェアや、所与のプログラム（ゲームプログラム）により実現できる。

【００３２】

操作部１６０は、プレーヤが操作データを入力するためのものであり、その機能は、レバー、ボタン、筐体などのハードウェアにより実現できる。

【００３３】

記憶部１７０は、処理部１００や通信部１９６などのワーク領域となるもので、その機能はＲＡＭなどのハードウェアにより実現できる。

【００３４】

情報記憶媒体（コンピュータにより使用可能な記憶媒体）１８０は、プログラムやデータなどの情報を格納するものであり、その機能は、光ディスク（ＣＤ、ＤＶＤ）、光磁気ディスク（ＭＯ）、磁気ディスク、ハードディスク、磁気テープ、或いはメモリ（ＲＯＭ）などのハードウェアにより実現できる。処理部１００は、この情報記憶媒体１８０に格納される情報に基づいて本発明（本実施形態）の種々の処理を行う。即ち情報記憶媒体１８０には、本発明（本実施形態）の手段（特に処理部１００に含まれるブロック）を実行するための情報（プログラム或いはデータ）が格納される。

【００３５】

なお、情報記憶媒体１８０に格納される情報の一部又は全部は、システムへの電源投入時等に記憶部１７０に転送されることになる。また情報記憶媒体１８０に記憶される情報は、本発明の処理を行うためのプログラムコード、画像データ、音データ、表示物の形状データ、テーブルデータ、リストデータ、本発明の処理を指示するための情報、その指示に従って処理を行うための情報等の少なくとも１つを含むものである。

【００３６】

表示部１９０は、本実施形態により生成された画像を出力するものであり、その機能は、ＣＲＴ、ＬＣＤ、或いはＨＭＤ（ヘッドマウントディスプレイ）などのハードウェアにより実現できる。

【００３７】

音出力部１９２は、本実施形態により生成された音を出力するものであり、その機能は、スピーカなどのハードウェアにより実現できる。

【００３８】

携帯型情報記憶装置１９４は、プレーヤの個人データ（セーブデータ）などが記憶されるものであり、この携帯型情報記憶装置１９４としては、メモリカードや携帯型ゲーム装置などを考えることができる。

【００３９】

通信部１９６は、外部（例えばホスト装置や他のゲームシステム）との間で通信を行うための各種の制御を行うものであり、その機能は、各種プロセッサ、或いは通信用ＡＳＩＣなどのハードウェアや、プログラムなどにより実現できる。

【００４０】

なお本発明（本実施形態）の手段を実行するためのプログラム或いはデータは、ホスト装置（サーバー）が有する情報記憶媒体からネットワーク及び通信部１９６を介して情報記憶媒体１８０に配信するようにしてもよい。このようなホスト装置（サーバー）の情報記憶媒体の使用も本発明の範囲内に含まれる。

【００４１】

処理部１００は、ゲーム処理部１１０、画像生成部１３０、音生成部１５０を含む。

【００４２】

ここでゲーム処理部１１０は、コイン（代価）の受け付け処理、各種モードの設定処理、ゲームの進行処理、選択画面の設定処理、オブジェクト（１又は複数のプリミティブ面）の位置や回転角度（Ｘ、Ｙ又はＺ軸回り回転角度）を求める処理、オブジェクトを動作させる処理（モーション処理）、視点の位置（仮想カメラの位置）や視線角度（仮想カメラの回転角度）を求める処理、マップオブジ

エクトなどのオブジェクトをオブジェクト空間へ配置するための処理、ヒットチェック処理、ゲーム結果（成果、成績）を演算する処理、複数のプレーヤが共通のゲーム空間でプレイするための処理、或いはゲームオーバー処理などの種々のゲーム処理を、操作部１６０からの操作データや、携帯型情報記憶装置１９４からの個人データや、ゲームプログラムなどに基づいて行う。

【００４３】

画像生成部１３０は、ゲーム処理部１１０からの指示等にしたがって各種の画像処理を行い、例えばオブジェクト空間内で仮想カメラ（視点）から見える画像を生成して、表示部１９０に出力する。また、音生成部１５０は、ゲーム処理部１１０からの指示等にしたがって各種の音処理を行い、ＢＧＭ、効果音、音声などの音を生成し、音出力部１９２に出力する。

【００４４】

なお、ゲーム処理部１１０、画像生成部１３０、音生成部１５０の機能は、その全てをハードウェアにより実現してもよいし、その全てをプログラムにより実現してもよい。或いは、ハードウェアとプログラムの両方により実現してもよい。

【００４５】

ゲーム処理部１１０は、移動・動作演算部１１２を含む。

【００４６】

ここで移動・動作演算部１１２は、車などのオブジェクトの移動情報（位置データ、回転角度データ）や動作情報（オブジェクトの各パーツの位置データ、回転角度データ）を演算するものであり、例えば、操作部１６０によりプレーヤが入力した操作データやゲームプログラムなどに基づいて、オブジェクトを移動させたり動作させたりする処理を行う。

【００４７】

より具体的には、移動・動作演算部１１２は、オブジェクトの位置や回転角度を例えば1フレーム（１／６０秒）毎に求める処理を行う。例えば（ｋ－１）フレームでのオブジェクトの位置をＰＭk-1、速度をＶＭk-1、加速度をＡＭk-1、１フレームの時間を△ｔとする。するとｋフレームでのオブジェクトの位置ＰＭ

k、速度ＶＭkは例えば下式（１）、（２）のように求められる。

【００４８】

$$PMk = PMk-1 + VMk-1 \times \Delta t \quad (1)$$

$$VMk = VMk-1 + AMk-1 \times \Delta t \quad (2)$$

画像生成部１３０は、ジオメトリ処理部１３２、インデックス番号設定部１３４、描画部１４０を含む。

【００４９】

ここで、ジオメトリ処理部１３２は、座標変換、クリッピング処理、透視変換、或いは光源計算などの種々のジオメトリ処理（３次元演算）を行う。そして、ジオメトリ処理後（透視変換後）のオブジェクトデータ（オブジェクトの頂点座標などの形状データ、或いは頂点テクスチャ座標、輝度データ等）は、記憶部１７０の主記憶領域１７２に保存される。

【００５０】

インデックス番号設定部１３４は、元画像の画像情報（例えば透視変換後の画像の情報）を、ＬＵＴ（ルックアップテーブル）記憶部１７８に記憶されるインデックスカラー・テクスチャマッピング用のＬＵＴのインデックス番号として設定するための処理を行う。ここで、画像情報としては、例えば、色情報（ＲＧＢ、ＹＵＶ等）、α値（各画素に関連づけられて記憶される情報であり色情報以外のプラスアルファの情報）、奥行き値（Ｚ値）等、種々の情報を考えることができる。

【００５１】

そして本実施形態では、元画像の各画素の奥行き値がインデックス番号として設定されたルックアップテーブルを用いて、仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行うことで、元画像の各画素の奥行き値に応じた値に各画素のα値を設定する。これにより、いわゆる被写界深度の表現が可能になる。

【００５２】

なお、仮想カメラの焦点から遠い画素ほどα値が大きくなるように（広義には、ぼかし画像の合成比率が高くなるように）、ＬＵＴ（ルックアップテーブル）

における各画素の奥行き値（インデックス番号）とα値との対応関係を設定することが望ましい。また、ＬＵＴにおける各画素の奥行き値とα値との対応関係を変化させることで、被写界深度の範囲やぼかしエフェクトの強弱を可変に制御してもよい。

【００５３】

描画部１４０は、ジオメトリ処理後のオブジェクト（モデル）を、描画領域１７４（フレームバッファ、別バッファ等のピクセル単位で画像情報を記憶できる領域）に描画するための処理を行うものであり、テクスチャマッピング部１４２、α合成部１４４、陰面消去部１４６を含む。

【００５４】

ここでテクスチャマッピング部１４２は、テクスチャ記憶部１７６に記憶されるテクスチャをオブジェクトにマッピングするための処理（オブジェクトにマッピングするテクスチャを指定する処理、テクスチャを転送する処理等）を行う。この場合、テクスチャマッピング部１４２は、ＬＵＴ記憶部１７８に記憶されるインデックスカラー・テクスチャマッピング用のＬＵＴ（ルックアップテーブル）を用いたテクスチャマッピングを行うことができる。

【００５５】

そして本実施形態では、テクスチャマッピング部１４２が、元画像の画像情報がインデックス番号として設定されたＬＵＴを用いて、仮想オブジェクト（表示画面サイズのポリゴン、分割ブロックサイズのポリゴン等）に対してテクスチャマッピングを行う。これにより、奥行き値（Ｚ値）をＮビット化する処理や、奥行き値をα値に変換する処理や、種々の画像変換処理（ガンマ補正、ネガポジ反転、ポスタリゼーション、ソラリゼーション、２値化、モノトーンフィルタ、セピアフィルタ）を少ない処理負担で実現できるようになる。

【００５６】

また本実施形態では、元画像とα合成（αブレンディング、α加算、α減算、半透明処理等）されるぼかし画像（最もぼけた画像）の生成処理を、テクセル補間方式（バイリニアフィルタ方式、トライリニアフィルタ方式）のテクスチャマッピングを有効利用して実現している。

【００５７】

即ち、テクスチャマッピング部１４２は、テクスチャとして設定された元画像を、テクスチャ座標を例えば１ピクセル（テクセル）よりも小さい値だけシフトさせながら（例えば元画像の描画位置に基づき得られるテクスチャ座標からシフトさせながら）、テクセル補間方式で仮想オブジェクト（ぼかし領域と同一形状のオブジェクト）にマッピングする。このようにすれば、テクスチャ座標をシフトさせるだけという簡素な処理で、元画像の合成対象となるぼかし画像を生成できるようになる。

【００５８】

α合成部１４４は、元画像とそのぼかし画像とを、描画領域１７４（フレームバッファ等）の各画素に対して設定されたα値（Ａ値）に基づいて合成する処理を行う。例えばα合成がαブレンディングである場合には、下式のように元画像とぼかし画像とが合成される。

【００５９】

$$R_Q=(1-\alpha)\times R_1+\alpha\times R_2 \qquad (3)$$

$$G_Q=(1-\alpha)\times G_1+\alpha\times G_2 \qquad (4)$$

$$B_Q=(1-\alpha)\times B_1+\alpha\times B_2 \qquad (5)$$

ここで、R_1、G_1、B_1は、描画領域１７４に既に描画されている元画像の色（輝度）のＲ、Ｇ、Ｂ成分であり、R_2、G_2、B_2は、ぼかし画像の色のＲ、Ｇ、Ｂ成分である。またR_Q、G_Q、B_Qは、αブレンディングにより生成されるＲ、Ｇ、Ｂ成分である。

【００６０】

陰面消去部１４６は、Ｚ値（奥行き値）が格納されるＺバッファ１７９（Ｚプレーン）を用いて、Ｚバッファ法のアルゴリズムにしたがった陰面消去を行う。本実施形態では、このＺバッファ１７９に書き込まれたＺ値をα値に変換し、そのα値に基づいて元画像とぼかし画像の合成処理を行っている。

【００６１】

なお、本実施形態のゲームシステムは、１人のプレーヤのみがプレイできるシングルプレーヤモード専用のシステムにしてもよいし、このようなシングルプレ

ーヤモードのみならず、複数のプレーヤがプレイできるマルチプレーヤモードも備えるシステムにしてもよい。

【００６２】

また複数のプレーヤがプレイする場合に、これらの複数のプレーヤに提供するゲーム画像やゲーム音を、１つの端末を用いて生成してもよいし、ネットワーク（伝送ライン、通信回線）などで接続された複数の端末（ゲーム装置、携帯電話）を用いて生成してもよい。

【００６３】

２．本実施形態の特徴

２．１　インデックスカラー・テクスチャマッピングの利用

さて、ゲームシステムにおいて画像を生成する際には、モニタ（表示部）の非線形特性を補正するために、画像に対してガンマ補正と呼ばれる変換を施すことが望ましい。

【００６４】

そして、このようなガンマ補正を実現する手法としては、以下に説明する第１、第２の手法がある。

【００６５】

第１の手法では、図２（Ａ）に示すように、メインメモリ８０２上にガンマ補正用のＬＵＴ（ルックアップテーブル）を用意しておく。そして、ＣＰＵ８００（ＣＰＵ上で動作するソフトウェア）は、ＶＲＡＭ８０６内にあるフレームバッファ８０８から、元画像の各画素の色情報（ＲＧＢ）を読み出す。そして、読み出された色情報に基づいてガンマ補正用ＬＵＴを参照し、ガンマ補正後の色情報を得る。次に、得られたガンマ補正後の色情報をフレームバッファの対応する画素に書き戻す。そして、以上の処理を、元画像の全ての画素に対して行う。

【００６６】

一方、第２の手法では、図２（Ｂ）に示すように、ＣＰＵ８１０の制御下で動作する描画プロセッサ８１２の後段に、ハードウェアによりガンマ補正を実現するガンマ補正回路８１４を設ける。そして、描画プロセッサ８１２により生成された色情報に対して、ガンマ補正回路８１４がガンマ補正を施し、モニタ８１６

に出力する。

【００６７】

しかしながら、図２（Ａ）の第１の手法では、フレームバッファ８０８からの色情報の読み出し、ガンマ補正用ＬＵＴ８０４の参照、ガンマ補正用ＬＵＴ８０４からの色情報の読み出し、読み出した色情報のフレームバッファ８０８への書き戻しなどの全ての処理を、ＣＰＵ８００上で動作するソフトウェアが行うことになる。従って、処理の高速化を図れず、表示画面の全画素に対するガンマ補正を、１フレーム内で完了するのは困難となる。また、ＣＰＵ８００の処理負荷が非常に重くなり、他の処理に悪影響を及ぼすという問題も招く。

【００６８】

一方、図２（Ｂ）の第２の手法では、専用のハードウェアであるガンマ補正回路８１４が使用されるため、高速なガンマ補正が可能になる。従って、表示画面の全画素に対するガンマ補正を１フレーム内で完了することも容易となる。また、ＣＰＵ８１０の処理負荷も少ないため、他の処理に悪影響が及ぶ問題も解決できる。

【００６９】

しかしながら、図２（Ｂ）の第２の手法では、専用のハードウェアであるガンマ補正回路８１４が別途必要になってしまう。従って、ゲームシステムが大規模化し、製品コストの増加の問題を招く。

【００７０】

特に、家庭用ゲームシステムにおいては、製品の普及化を図るために、低コスト化が厳しく要求されており、ほとんどの家庭用ゲームシステムでは、図２（Ｂ）に示すようなガンマ補正回路がハードウェアとして設けられていない。従って、ガンマ補正を実現するためには、図２（Ａ）のような第１の手法を採用せざるを得ない。

【００７１】

ところが、前述のように第１の手法では、１フレーム内で全表示画面分のガンマ補正を完了するのは困難であり、他の処理にも悪影響を及ぼす。従って、家庭用ゲームシステムにおいては、ガンマ補正の実施自体を断念せざるを得なかった

【００７２】

そこで、本出願の発明者は、インデックスカラー・テクスチャマッピングにおいて使用されるルックアップテーブルＬＵＴの存在に着目した。

【００７３】

即ち、インデックスカラーテクスチャーマッピングでは、テクスチャ記憶部の使用記憶容量を節約するために、図３のＡ１に示すように、実際の色情報（ＲＧＢ）ではなくインデックス番号が、テクスチャの各テクセルに関連づけて記憶される。また、図３のＡ２に示すように、インデックスカラー・テクスチャマッピング用のＬＵＴ（カラーパレット）には、インデックス番号により指定される色情報が記憶される。そして、オブジェクトに対してテクスチャマッピングを行う際には、テクスチャの各テクセルのインデックス番号に基づいてＬＵＴを参照し、対応する色情報をＬＵＴから読み出し、読み出された色情報をフレームバッファに描画する。

【００７４】

このようなインデックスカラーモードのテクスチャマッピングでは、ＬＵＴを用いない通常モードのテクスチャマッピングに比べて、使用できる色数は少なくなる（例えば２５６色）。しかしながら、テクスチャ記憶部に実際の色情報（例えば１６ビットの色情報）を記憶する必要が無くなるため、テクスチャ記憶部の使用記憶容量を大幅に節約できる。

【００７５】

本実施形態は、このようなインデックスカラー・テクスチャマッピングを通常とは異なる形態で利用している点に特徴がある。

【００７６】

即ち、まず図４のＢ１に示すように、フレームバッファ（広義には描画領域）に描画されている元画像の各画素の画像情報（例えば色情報）を、ガンマ補正用のルックアップテーブルＬＵＴのインデックス番号として設定する（インデックス番号とみなす）。そしてＢ２に示すように、元画像の画像情報がインデックス番号として設定されたＬＵＴを用いて、仮想オブジェクト（例えば表示画面サイ

ズのポリゴン）に対してインデックスカラー・テクスチャマッピングを行い、元画像の画像情報を変換する。そしてＢ３に示すように、変換後の画像情報を、フレームバッファ（描画領域）などに描き戻す。

【００７７】

以上のようにして本実施形態では、図５（Ａ）に示すような元画像から、図５（Ｂ）に示すようなガンマ補正が施された画像を得ることに成功している。即ち図５（Ｂ）の画像では、図５（Ａ）に比べて、よりコントラストのはっきりした画像になっている。

【００７８】

例えば図２（Ａ）の第１の手法では、元画像の色情報の読み出し、ガンマ補正用ＬＵＴの参照、色情報のフレームバッファへの書き戻しなどの全ての処理をＣＰＵ上で動作するソフトウェアが行うことになるため、処理の高速化を図れないと共に、ＣＰＵの処理負荷も過大になる。

【００７９】

これに対して本実施形態では、インデックスカラー・テクスチャマッピングを有効利用してガンマ補正を実現しており、このインデックスカラー・テクスチャマッピングは、専用のハードウェアである描画プロセッサ（描画部）により高速に実行される。従って本実施形態によれば、図２（Ａ）の第１の手法に比べて高速にガンマ補正を実行でき、全表示画面分のガンマ補正を１フレーム（例えば１／６０秒、１／３０秒）内で完了することも容易となる。

【００８０】

また、インデックスカラー・テクスチャマッピングは、メインプロセッサ（ＣＰＵ）とは独立に動作する描画プロセッサにより実行できるため、メインプロセッサ（ＣＰＵ）の処理負荷の増加も最小限に抑えることができる。従って、ガンマ補正の実行が要因となって他の処理に悪影響が及ぶ事態も防止できる。

【００８１】

また、従来のゲームシステムでは、描画プロセッサの処理能力はそれほど高くなかった。従って、フレームバッファへの元画像の描画と、表示画面サイズのポリゴンの描画を、１フレーム内で完了させることは難しかった。

【００８２】

しかしながら、ゲームシステムにおいて、描画プロセッサの処理能力の向上は、他の回路ブロックの処理能力の向上に比べて著しく大きく、非常に高いフィルレート（１秒間にレンダリングできるテクセル数）を持つ描画プロセッサがゲームシステムに使用されるようになってきた。従って、フレームバッファへの元画像の描画と、表示画面サイズのポリゴンの描画を、１フレーム内で完了させることも容易となり、インデックスカラー・テクスチャマッピングを有効利用したガンマ補正も無理なく実現できるようになってきた。

【００８３】

また図２（Ｂ）の第２の手法では、専用のハードウェアであるガンマ補正回路が別途必要になり、ゲームシステムの高コスト化を招く。また、このようなガンマ補正回路を元々有しない家庭用ゲームシステムなどでは、図２（Ｂ）に示す第２の手法を実現することはできず、図２（Ａ）の手法を採用せざるを得なかった。

【００８４】

これに対して本実施形態では、インデックスカラー・テクスチャマッピングを有効利用してガンマ補正を実現しており、このインデックスカラー・テクスチャマッピングは、描画プロセッサが元々持っているハードウェアにより実行される。従って本実施形態によれば、図２（Ｂ）のようなガンマ補正回路を新たに付加する必要がなく、ゲームシステムが高コスト化してしまう事態を防止できる。また、ガンマ補正回路を元々有していない家庭用ゲームシステムにおいても、ハードウェアによる高速なガンマ補正を実現できるようになる。

【００８５】

なお、図４では表示画面サイズのポリゴンにテクスチャマッピングしてガンマ補正（ビデオフィルタ）を実現しているが、表示画面を分割したブロックのサイズのポリゴンにテクスチャマッピングするようにしてもよい。

【００８６】

即ち、図６のＣ１に示すように、フレームバッファ上の元画像（表示画面）を複数のブロックに分割し、Ｃ２に示すように、各ブロックの画像を、ＬＵＴを用

いて分割ブロックサイズのポリゴンにテクスチャマッピングする。そして、得られた分割ブロックサイズの画像をフレームバッファ（描画領域）に描き戻す。

【００８７】

或いは、透視変換後（スクリーン座標系への変換後）のオブジェクトの画像の全部又は一部を内包し、透視変換後のオブジェクトの大きさに応じてその大きさが変化するようなポリゴン（仮想オブジェクト）を生成し、そのポリゴンにテクスチャマッピングする。

【００８８】

このようにすれば、例えばテクスチャマッピングされたポリゴンを別バッファに一時的に描画するような場合に、ＶＲＡＭ上での別バッファの占有領域を小さくできる。

【００８９】

即ち、図４のように表示画面サイズのポリゴンにテクスチャマッピングすると、この表示画面サイズのポリゴンを一時的に描画するために、表示画面サイズの別バッファをＶＲＡＭ上に確保しなければならず、他の処理に支障を来すおそれがある。

【００９０】

図６のように、分割ブロックサイズのポリゴンにテクスチャマッピングするようにすれば、ＶＲＡＭ上には分割ブロックサイズの別バッファを用意すれば済むため、別バッファの占有領域を小さくできる。従って、限られたハードウェア資源を有効利用することが可能になる。

【００９１】

２．２　マスク処理

さて、ガンマ補正では、１つの入力値（ＲＩＮ、ＧＩＮ又はＢＩＮ）に対して１つの値（ＲＯＵＴ、ＧＯＵＴ又はＢＯＵＴ）が出力されるようなＬＵＴが必要になる。

【００９２】

ところが、図３に示すインデックスカラー・テクスチャマッピング用のＬＵＴは、元々、ガンマ補正用に設計されたものではないため、１つ入力値（インデッ

クス番号）に対して、複数の値（例えばＲＯＵＴ、ＧＯＵＴ、及びＢＯＵＴ）が出力されてしまう。従って、このＬＵＴの不整合を解決しなければならないという課題がある。

【００９３】

そこで本実施形態では、元画像の画像情報（Ｒ、Ｇ、Ｂ、Ｚ値又はα値等）をＬＵＴのインデックス番号として設定する場合において、変換により得られた画像情報のうち必要な画像情報のみが描画領域（フレームバッファ、別バッファ）に描画され、他の画像情報が描画されないようにするためのマスク処理を行っている。

【００９４】

より具体的には、図７のＤ１に示すように、元画像のＲプレーンの値をインデックス番号に設定して、ＬＵＴを用いたテクスチャマッピングを行うと、Ｒ（ＲＯＵＴ）、Ｇ（ＧＯＵＴ）、Ｂ（ＢＯＵＴ）という３つプレーンの値が出力される。そして、この場合には、Ｄ２に示すように、出力されたＲプレーンの値のみを描画領域に描画し、Ｇプレーン、Ｂプレーンの値についてはマスクして描画領域に描画されないようにする。

【００９５】

また、図７のＤ３に示すように、元画像のＧプレーンの値をインデックス番号に設定してテクスチャマッピングを行った場合には、Ｄ４に示すように、出力されたＧプレーンの値のみを描画領域に描画し、Ｒプレーン、Ｂプレーンの値についてはマスクして描画領域に描画されないようにする。

【００９６】

また、図７のＤ５に示すように、元画像のＢプレーンの値をインデックス番号に設定してテクスチャマッピングを行った場合には、出力されたＢプレーンの値のみを描画領域に描画し、Ｒプレーン、Ｇプレーンの値についてはマスクして描画領域に描画されないようにする。

【００９７】

以上のようにすることで、元々はガンマ補正用に設計されていないインデックスカラー・テクスチャマッピング用のＬＵＴを用いながらも、少ない処理負荷で

、元画像の画像変換処理を実行できるようになる。

【００９８】

２．３　Ｚ値、α値への利用

以上では、インデックスカラー・テクスチャマッピング用のＬＵＴに基づき出力される色情報Ｒ、Ｇ、Ｂを利用する場合について説明した。

【００９９】

しかしながら、インデックスカラー・テクスチャマッピング用のＬＵＴに基づき出力されるα値（Ａ値。画素に関連づけて設定される色情報以外の情報）を利用するようにしてもよい。

【０１００】

例えば図８に示すように、Ｒプレーン（又はＧ、Ｂ）の値をインデックス番号として設定して、ＬＵＴを用いたテクスチャマッピングを行い、α（αＯＵＴ）プレーンを生成する。そして、生成されたαプレーンを用いて、マスク処理などを行うようにする。

【０１０１】

即ち、例えばＲ値が０～１２７の時にはα値（αＯＵＴ）が０になり、Ｒ値が１２８～２５５の時にはα値が２５５になるようにα値が設定されたＬＵＴを使用する。そして、α値が２５５よりも小さい画素についてはマスク処理を行わず、α値が２５５の画素についてはマスク処理を行うようにする。このようにすれば、Ｒ値が１２８以上の画素に対してだけマスク処理が行われるようになり、各画素のＲ値の大小に応じたマスク処理を行うことができるようになる。

【０１０２】

なお、生成されたαプレーンの値をα合成の係数（透明度、半透明度、不透明度）として使用してもよい。

【０１０３】

また、ＬＵＴのインデックス番号として設定される画像情報は色情報に限定されない。即ち、描画領域（ＶＲＡＭ）上にあり、ＬＵＴのインデックス番号として設定できる画像情報であればよい。

【０１０４】

例えば図９に示すように、Ｚ値（奥行き値）をＬＵＴのインデックス番号として設定するようにしてもよい。

【０１０５】

そして、この場合には、Ｚ値をインデックス番号に設定してインデックスカラー・テクスチャマッピングを行うことで得られるαプレーンの値を、例えばα合成の係数として使用するようにする。このようにすることで、Ｚ値に応じた値のα値を設定できるようになり、ぼかし画像を用いた被写界深度などの表現が可能になる。

【０１０６】

即ち図９に示すようなＬＵＴを用いたテクスチャマッピングを行うことで、図１０のＦ１に示すように、元画像の各画素Ａ、Ｂ、Ｃ、ＤのＺ値ＺＡ、ＺＢ、ＺＣ、ＺＤに応じた値に各画素のα値αＡ、αＢ、αＣ、αＤを設定する。そして、例えば図１０のＦ２に示すようなαプレーンを生成する。より具体的には、仮想カメラ１０の焦点（注視点）から遠い画素（焦点とのＺ値の差が大きい画素）ほど、例えば大きなα値を設定する。これにより、仮想カメラ１０の焦点から遠い画素ほど、ぼかし画像の合成比率が高くなる。

【０１０７】

そして、図１０のＦ３に示すように、生成されたαプレーン（各画素に設定されたα値）に基づいて、元画像とぼかし画像のα合成（αブレンディング等）を行う。なお、図１１（Ａ）に元画像の例を示し、図１１（Ｂ）に、そのぼかし画像の例を示す。

【０１０８】

このように、Ｚ値（奥行き値）に応じて設定されたα値に基づき元画像（図１１（Ａ））とぼかし画像（図１１（Ｂ））のα合成を行うことで、例えば、仮想カメラの焦点（ピントが合っている点として設定される点）から遠くなるほどぼけて見える画像を生成できるようになり、いわゆる被写界深度の表現が可能になる。これにより、画面内の全ての被写体にピントが合っていた従来のゲーム画像とは異なり、現実世界の視界画像のように視点からの距離に応じてフォーカシングされたリアルで自然なゲーム画像を生成できる。この結果、プレーヤの仮想現

実感を格段に向上できる。

【０１０９】

例えば図１２に、Ｚ値に応じたα値の設定の一例を示す。なお、図１２では、α値は、その大きさが１．０以下になるように正規化されている。

【０１１０】

そして図１２では、Ｚ値Ｚ１～Ｚ４、Ｚ１’～Ｚ４’（しきい値）により領域ＡＲ０～ＡＲ４、ＡＲ１’～ＡＲ４’の区分けが行われる。そして、これらの領域ＡＲ０～ＡＲ４、ＡＲ１’～ＡＲ４’に対して、α値α０～α４、α１’～α４’が設定される。

【０１１１】

例えば、Ｚ１～Ｚ２の間の領域ＡＲ１にある画素については、そのα値がα１に設定され、Ｚ２～Ｚ３の間の領域ＡＲ２にある画素については、そのα値がα２に設定される。また、Ｚ１’～Ｚ２’の間の領域ＡＲ１’にある画素については、そのα値がα１’に設定され、Ｚ２’～Ｚ３’の間の領域ＡＲ２’にある画素については、そのα値がα２’に設定される。

【０１１２】

そして、各領域に設定されるα値には例えば以下の関係式が成り立つ。

【０１１３】

$$\alpha 0 < \alpha 1 < \alpha 2 < \alpha 3 < \alpha 4 \qquad (6)$$

$$\alpha 0 < \alpha 1' < \alpha 2' < \alpha 3' < \alpha 4' \qquad (7)$$

これらの式（６）、（７）から明らかなように、仮想カメラ１０の焦点（注視点）から遠いほどα値が大きくなっている。即ち、仮想カメラ１０の焦点とのＺ値の差が大きい画素ほど、ぼかし画像の合成比率が高くなるように、α値が設定されている。

【０１１４】

このようにα値を設定することで、仮想カメラの焦点から遠くなるほどぼけて見える画像を生成でき、いわゆる被写界深度の表現が可能になる。

【０１１５】

しかも本実施形態では、ＬＵＴを用いた１回のテクスチャマッピングで、各画

素のＺ値をα値に変換できるため、処理負担が非常に軽いという利点がある。

【０１１６】

即ち、ＬＵＴを用いないα値の設定手法の例として、図１３（Ａ）、（Ｂ）、（Ｃ）に示す手法が考えられる。

【０１１７】

この手法では図１３（Ａ）に示すように、まず、Ｚ値がＺ１に設定される仮想オブジェクトＯＢ１（ポリゴン）を、フレームバッファに描画することで、仮想オブジェクトＯＢ１を基準に奥側にある画素のα値を更新する。即ち、Ｚ値に基づく陰面消去手法を有効利用して、ＯＢ１よりも奥側にある画素のα値を更新する。

【０１１８】

次に、図１３（Ｂ）に示すように、Ｚ値がＺ２に設定される仮想オブジェクトＯＢ２をフレームバッファに描画することで、仮想オブジェクトＯＢ２を基準に奥側にある画素のα値を更新する。同様に、図１３（Ｃ）に示すように、Ｚ値がＺ３に設定される仮想オブジェクトＯＢ３をフレームバッファに描画することで、仮想オブジェクトＯＢ３を基準に奥側にある画素のα値を更新する。

【０１１９】

このような手法を採用すれば、領域ＡＲ１にある画素のα値をα１に設定し、領域ＡＲ２にある画素のα値をα２に設定し、領域ＡＲ３にある画素のα値をα３に設定できるようになる。即ち、ＬＵＴを用いなくても、各画素のＺ値に応じた値に各画素のα値を設定できる。

【０１２０】

しかしながら、この手法では、Ｚ値のしきい値の段階数分だけ、仮想オブジェクトの描画処理が必要になる。例えば図１２の場合は８回の描画処理が必要になる。従って、この手法には描画処理の負担が重くなるという欠点がある。一方、描画処理の負担を軽減するためにＺ値のしきい値の段階数を減らすと、今度は、Ｚ値のしきい値の境界が表示画面上で帯状に見えてしまう事態が生じ、表示品質が低下する。

【０１２１】

ＬＵＴを用いてＺ値をα値に変換する本実施形態の手法によれば、ＬＵＴを用いた１回のテクスチャマッピングで、各画素のＺ値をα値に一斉に変換できる。例えば、ＬＵＴのインデックス番号（エントリー）のビット数が８ビットの場合には、ＬＵＴを用いた１回のテクスチャマッピングで、２５６段階のＺ値のしきい値で区分けされるα値を得ることができる。従って、Ｚ値のしきい値の境界が表示画面上で帯状に見えてしまう事態を防止でき、少ない処理負担で高品質な画像を生成できる。

【０１２２】

２．４　Ｚ値の８ビット化

さて、ゲームシステムにおいては、陰面消去の精度を高めるために、Ｚ値のビット数は例えば２４ビット、３２ビットというように非常に多い。

【０１２３】

一方、インデックスカラー・テクスチャマッピング用のＬＵＴのインデックス番号のビット数（ＬＵＴのエントリーのビット数）は、例えば８ビットというようにＺ値のビット数に比べて少ない。

【０１２４】

従って、図９のようにインデックスカラー・テクスチャマッピング用のＬＵＴを用いてＺ値をα値に変換する場合には、その前処理として、Ｚ値を、ＬＵＴのインデックス番号のビット数と同じビット数のＺ値（以下、Ｚ２値と呼ぶ）に変換する処理が必要になる。即ち、ＬＵＴのインデックス番号のビット数が８ビットである場合には、Ｚ値を８ビットのＺ２値（第２の奥行き値）に変換する処理が必要になる。

【０１２５】

この場合に、矛盾の無いＺ２値を得るためには、Ｚ値の最上位ビットを含む上位の８ビットをビットセレクトしたものをＺ２値とする必要がある。即ち図１４において、ビット２３（最上位ビット）～１６の８ビットをビットセレクトして、Ｚ２値として設定する。

【０１２６】

しかしながら、このようにＺ値の上位ビット２３～１６の８ビットをＺ２値に

設定し、このＺ２値をＬＵＴを用いてα値に変換すると、α値の区分け数が少なくなってしまうということが判明した。

【０１２７】

例えば図１５のように、Ｚ値が４ビットであり、Ｚ値の上位の２ビットをセレクトしてＺ２値に設定し、このＺ２値をα値に変換した場合を考える。この場合には、Ｚ＝０～１５の全範囲において４段階のしきい値でα値が区分けされることになる。

【０１２８】

しかしながら、図１５のＯＢ１のようにスクリーンＳＣ（透視変換面）の手前にあるオブジェクトについては、通常はニアクリップされてしまう。そして、例えば図１５においてＺ＝１０～１５の範囲にあるオブジェクトがニアクリップされるような場合には、Ｚ値の最上位ビットに１が立つことは希となる。また、ＯＢ２のようにスクリーンＳＣの近くにあるオブジェクトについては、何れにせよ最もぼけた画像になるため、ぼけ具合を精度良く制御する必要がない。従って、図１５のＧ１に示す部分における２段階の区分けは無駄になる。

【０１２９】

そこで本実施形態では図１４に示すように、元画像の各画素のＺ値（奥行き値）を、Ｚ値の最上位ビットよりも下位のビットＩ～Ｊ（例えばビット１９～１２）により構成されるＺ２値（第２の奥行き値）に変換し、このＺ２値を、インデックスカラー・テクスチャマッピング用のＬＵＴのインデックス番号として設定してテクスチャマッピングを行い、各画素のα値を求める。そして、この求められた各画素のα値に基づいて、元画像とぼかし画像をα合成する。

【０１３０】

このようにすれば図１６に示すように、仮想カメラ１０の焦点（注視点）付近にあるオブジェクト（例えばＯＢ３、ＯＢ４）についてだけ、多段階（図１６では４段階）のしきい値で区分けされたα値により、ぼけ具合を精度良く制御できるようになる。従って、生成される画像の品質を高めることができる。

【０１３１】

そして本実施形態では、Ｚ値のビットＩ～Ｊ以外のビットの値に応じて、Ｚ２

値を所与の値にクランプするようにしている。より具体的には図１６に示すように、Ｚ値の上位のビットに例えば１が立った場合には、Ｚ２値を最大値（広義には所与の値）にクランプする。このようにすれば、ＯＢ１やＯＢ２のように、ぼけ具合を精度良く制御する必要がないオブジェクトについては、Ｚ２値が最大値に設定され、最もぼけた画像に設定される。従って、Ｚ値を、そのビットＩ～Ｊにより構成されるＺ２値に変換しても、矛盾の無い画像を生成できる。

【０１３２】

２．５　ＬＵＴを利用したＺ値の変換処理

さて本実施形態では、図１４で説明したＺ値をＺ２値に変換する処理（８ビット化処理）を、インデックスカラー・テクスチャマッピング用のＬＵＴを利用したテクスチャマッピングにより実現している。即ち、Ｚ値をＬＵＴのインデックス番号として設定し、そのＬＵＴを用いて仮想オブジェクトに対してインデックスカラー・テクスチャマッピングを行い、Ｚ値をＺ２値に変換する。

【０１３３】

例えば図１７に示すように２４ビットのＺ値をＬＵＴを用いて変換する場合を考える。この場合には図１７のＨ１に示すように、Ｚ値のビット１５～８（ビットＭ～Ｎ）をＬＵＴ１（第１のルックアップテーブル）のインデックス番号に設定し、ＬＵＴ１を用いてインデックスカラー・テクスチャマッピングを行い、Ｚ値をＺ３値（第３の奥行き値）に変換する。

【０１３４】

次に図１７のＨ２に示すように、Ｚ値のビット２３～１６（ビットＫ～Ｌ）をＬＵＴ２（第２のルックアップテーブル）のインデックス番号に設定し、ＬＵＴ２を用いてインデックスカラー・テクスチャマッピングを行い、Ｚ値をＺ４値（第４の奥行き値）に変換する。

【０１３５】

そして図１７のＨ４に示すようにＺ３値とＺ４値とに基づいてＺ２値を求め、Ｈ５に示すようにこのＺ２値をＬＵＴ３（第３のルックアップテーブル）を用いてα値に変換する。

【０１３６】

より具体的には、ＬＵＴ１の変換により得られたＺ３値を描画領域（フレームバッファ、別バッファ）に描画する。その後、ＬＵＴ２の変換により得られたＺ４値を描画領域に描画する。この際、Ｚ３値の下位の４ビット（有効ビット）をマスクして、これらのビットにＺ４値が上書きされないようにして、Ｚ２値を求める。

【０１３７】

図１７の手法を採用することで、Ｚ値の任意の８ビット（広義には任意のビットＩ～Ｊ）を取り出すことが可能になる。

【０１３８】

即ち、ＬＵＴのインデックス番号に設定するためにＺ値の８ビットを取り出そうとした場合に、Ｚ値のビット２３～１６、１５～８又は７～０というように所定範囲の８ビットしか取り出せない場合がある。

【０１３９】

一方、Ｚ値の中のどの８ビットを取り出すかは、図１４、図１５で説明したように、ニアクリップの範囲や、仮想カメラの焦点（注視点）の位置などに応じて決められる。

【０１４０】

従って、Ｚ値のビット２３～１６、１５～８又は７～０というように所定範囲の８ビットしか取り出せないと、仮想カメラの焦点付近で最も精度良くぼけ具合を制御できるような適切なα値を得ることができない。

【０１４１】

例えば、Ｚ値のビット１９～１２をＺ２値として取り出せば、α値の実効的な区分け数（Ｚ値のしきい値数）を２５６段階にできる場合を考える。このような場合に、Ｚ値のビット２３～１６や１５～８しか取り出せないと、α値の実効的な区分け数が例えば１６段階になってしまい、画質が低下する。

【０１４２】

これに対して、図１７に示す手法を採用すれば、上記のようにＺ値の所定範囲の８ビットしか取り出せないような場合にも、Ｚ値の任意のビットＩ～ＪをＺ２値として取り出すことができるようになる。従って、ニアクリップの範囲や、仮

想カメラの焦点の位置などに応じて、α値を区分けするＺ値のしきい値の設定を最適なものにすることが可能になり、より高品質な画像を生成できるようになる。

【０１４３】

なお図１８、図１９、図２０（Ａ）に、ＬＵＴ１、ＬＵＴ２、ＬＵＴ３の具体例を示す。また図２０（Ｂ）に、Ｚ２値をα値に変換するＬＵＴ３の変換特性曲線の例を示す。

【０１４４】

図１８に示すようにＬＵＴ１は、インデックス番号として入力されたＺ値のビット１５～８（ビットＭ～Ｎ）を４ビットだけ右にシフトする変換を行う。例えば０ｘ１０（１６進数表現）は０ｘ０１に変換され、０ｘ２０は０ｘ０２に変換される。

【０１４５】

また図１９に示すようにＬＵＴ２は、インデックス番号として入力されたＺ値のビット２３～１６（ビットＫ～Ｌ）を４ビットだけ左にシフトする変換を行う。例えば０ｘ０１は０ｘ１０に変換され、０ｘ０２は０ｘ２０に変換される。

【０１４６】

そして図１９のＱ１に示すように、入力されたＺ値が０ｘ０Ｆよりも大きい場合には、ＬＵＴ２の出力は０ｘＦ０にクランプされる。

【０１４７】

Ｚ値がクランプされた場合の例を図２１に示す。図２１に示すように、ビット２０（ビットＩ～Ｊ以外のビット）に１が立った場合には、ＬＵＴ２の出力が０ｘＦ０にクランプされる。これによりＺ２値は０ｘＦ１になる。

【０１４８】

例えばＬＵＴ２の出力をクランプせずに、Ｚ値のビット１９～１２をそのまま取り出してしまうと、ビット２０に１が立っているにもかかわらず、Ｚ２値は０ｘ１１になってしまい、被写界深度の設定が誤った設定になってしまう問題が生じる。

【０１４９】

ＬＵＴ２の出力をクランプするようにすれば、このような問題が生じるのを防止でき、被写界深度の設定を適正なものにすることができる。

【０１５０】

しかも、このようにＬＵＴ２の出力をクランプしても、図１６から明らかなように、ニアクリップされるオブジェクトＯＢ１やスクリーンＳＣの近くのオブジェクトＯＢ２のぼけ度合いが最大値に設定されるだけであるため、画像が不自然になることもない。

【０１５１】

２．６　ぼかし画像の生成

さて、本実施形態では、テクスチャマッピングのバイリニアフィルタ方式（テクセル補間方式）を有効利用して、元画像（図１１（Ａ））の合成対象となるぼかし画像（図１１（Ｂ））を生成している。

【０１５２】

即ち、テクスチャマッピングにおいては画素の位置とテクセルの位置がずれる場合がある。

【０１５３】

この場合に、図２２に示すように、ポイントサンプリング方式では、画素（サンプリング点）Ｐの色ＣＰ（広義には画像情報）は、Ｐに最も距離が近いテクセルＴＡの色ＣＡになる。

【０１５４】

一方、バイリニアフィルタ方式では、Ｐの色ＣＰは、Ｐの周りのテクセルＴＡ、ＴＢ、ＴＣ、ＴＤの色ＣＡ、ＣＢ、ＣＣ、ＣＤを補間した色になる。

【０１５５】

より具体的には、ＴＡ～ＴＤの座標とＰの座標とに基づき、Ｘ軸方向の座標比β：１－β（０≦β≦１）と、Ｙ軸方向の座標比γ：１－γ（０≦γ≦１）を求める。

【０１５６】

この場合に、Ｐの色ＣＰ（バイリニアフィルタ方式での出力色）は、下式のようになる。

【０１５７】

$$CP=(1-\beta)\times(1-\gamma)\times CA+\beta\times(1-\gamma)\times CB+(1-\beta)\times\gamma\times CC+\beta\times\gamma\times CD \quad (8)$$

本実施形態では、このようにバイリニアフィルタ方式では色が自動的に補間されることに着目して、ぼかし画像を生成している。

【０１５８】

より具体的には図２３のＲ１に示すように、例えばフレームバッファに描画されている元画像をテクスチャとして設定する。そして、このテクスチャ（元画像）を仮想オブジェクトにバイリニアフィルタ方式でマッピングする際に、仮想オブジェクトの頂点に与えるテクスチャ座標を、例えば（０．５、０．５）だけ右下方向にシフト（ずらす、移動）させる。このようにすることで、バイリニアフィルタ方式の補間機能により自動的に、元画像の画素の色が周囲ににじんだようなぼかし画像を生成できるようになる。

【０１５９】

なお、画面全体をぼかす場合には、テクスチャ（元画像）をマッピングする仮想オブジェクトの形状は、画面（ぼかし領域）と同一形状に設定される。即ち、画面の頂点座標が（Ｘ、Ｙ）＝（０、０）、（６４０、０）、（６４０、４８０）、（０、４８０）であった場合には、仮想オブジェクトの頂点座標も（Ｘ、Ｙ）＝（０、０）、（６４０、０）、（６４０、４８０）、（０、４８０）になる。

【０１６０】

そして、この場合に、仮想オブジェクトの頂点ＶＸ１、ＶＸ２、ＶＸ３、ＶＸ４に与えるテクスチャ座標（Ｕ、Ｖ）を、各々、（０、０）、（６４０、０）、（６４０、４８０）、（０、４８０）に設定すれば、画面の画素の位置とテクスチャのテクセルの位置とがずれずに一致する。従って、画像はぼけない。

【０１６１】

これに対して、仮想オブジェクトの頂点ＶＸ１、ＶＸ２、ＶＸ３、ＶＸ４に与えるテクスチャ座標（Ｕ、Ｖ）を、各々、（０．５、０．５）、（６４０．５、０．５）、（６４０．５、４８０．５）、（０．５、４８０．５）に設定すれば、画面の画素の位置とテクスチャのテクセルの位置とがずれるようになる。従って

、バイリニアフィルタ方式の補間機能により、色の補間が行われ、画像がぼけて見えるようになる。

【０１６２】

なお、画面の一部の領域をぼかす場合には、仮想オブジェクトの形状を、そのぼかし領域と同一形状にすればよい。

【０１６３】

また本実施形態では、図２４のＲ３に示すように、元画像をテクスチャに設定し、例えば右下方向（第１のシフト方向）に０．５テクセルだけシフトしてバイリニアフィルタ方式でテクスチャマッピングを行い、第１のぼかし画像を生成する。次に、図２４のＲ４に示すように、この第１のぼかし画像をテクスチャに設定し、例えば左上方向（第２のシフト方向）に０．５テクセルだけシフトしてバイリニアフィルタ方式でテクスチャマッピングを行い、第２のぼかし画像を生成する。或いは、以上の処理（右下方向のシフトと左上方向のシフト）を複数回繰り返す。このようにすることで、更に自然でぼかし効果の強いぼかし画像を生成できるようになる。

【０１６４】

次に、バイリニアフィルタ方式の補間機能によりぼかし画像が生成される原理について説明する。

【０１６５】

例えば図２５（Ａ）に示すように、テクスチャ座標を０．５テクセルだけ右下方向にシフトさせて、バイリニアフィルタ方式のテクスチャマッピングを行ったとする。この場合には、上式（８）において$\beta=\gamma=1/2$になるため、テクセルＴ４４、Ｔ４５、Ｔ５４、Ｔ５５の色をＣ４４、Ｃ４５、Ｃ５４、Ｃ５５とすると、画素Ｐ４４の色ＣＰ４４は下式のようになる。

【０１６６】

ＣＰ４４＝（Ｃ４４＋Ｃ４５＋Ｃ５４＋Ｃ５５）／４　（９）

以上から明らかなように、図２５（Ａ）に示す変換により、テクセルＴ４４の色Ｃ４４（変換前の元画像の画素Ｐ４４の元の色に相当）は、周りの画素Ｐ３３、Ｐ３４、Ｐ４３、Ｐ４４に対して１／４ずつしみ出すことになる。

【０１６７】

そして、その後に図２５（Ｂ）に示すように、図２５（Ａ）で得られた画像をテクスチャとして、テクスチャ座標を０．５テクセルだけ左上方向にシフトさせてバイリニアフィルタ方式でテクスチャマッピングを行ったとする。この場合には、図２５（Ａ）の画素Ｐ３３、Ｐ３４、Ｐ４３、Ｐ４４が、図２５（Ｂ）のテクセルＴ３３、Ｔ３４、Ｔ４３、Ｔ４４に対応するようになる。そして、図２５（Ａ）でＰ３３、Ｐ３４、Ｐ４３、Ｐ４４（Ｔ３３、Ｔ３４、Ｔ４３、Ｔ４４）に対して１／４ずつしみ出した色Ｃ４４が、更に１／４倍されて周りの４つの画素に対してしみ出すことになる。即ち、結局、元のＴ４４の色Ｃ４４が１／４×１／４＝１／１６ずつ周りにしみ出すことになる。

【０１６８】

従って、図２５（Ａ）、（Ｂ）の変換により、画素Ｐ３３、Ｐ３４、Ｐ３５には、各々、色Ｃ４４（フレームバッファに描かれた元画像の画素Ｐ４４の元の色に相当）が１／１６、２／１６、１／１６ずつしみ出すことになる。また、画素Ｐ４３、Ｐ４４、Ｐ４５には、各々、色Ｃ４４が２／１６、４／１６、２／１６ずつしみ出し、画素Ｐ５３、Ｐ５４、Ｐ５５には、各々、色Ｃ４４が１／１６、２／１６、１／１６ずつしみ出すことになる。

【０１６９】

従って、図２５（Ａ）、（Ｂ）の変換により、結局、図２６（Ａ）に示すような平面フィルタが元画像に対して施されるようになる。この平面フィルタによれば、元画像の各画素の色がその周りに均一に広がるようになり、元画像の理想的なぼかし画像を生成できる。

【０１７０】

また、図２５（Ａ）、（Ｂ）の変換のセットを２回行えば、図２６（Ｂ）に示すような平面フィルタが元画像に対して施されるようになる。この平面フィルタによれば、図２６（Ａ）よりも更に理想的なぼかし画像を生成できる。

【０１７１】

３．本実施形態の処理

次に、本実施形態の処理の詳細例について、図２７、図２８のフローチャート

を用いて説明する。

【０１７２】

まず、元画像（透視変換後の画像）をフレームバッファに描画する（ステップＳ１）。この際に、Ｚバッファには各画素のＺ値が書き込まれることになる。

【０１７３】

次に、ＺバッファのＺ値のビット１５～８を変換するＬＵＴ１（図１８）、ビット２３～１６を変換するＬＵＴ２（図１９）、８ビット化されたＺ値をα値（Ａ値）に変換するＬＵＴ３（図２０（Ａ））を、ＶＲＡＭに転送する（ステップＳ２）。

【０１７４】

次に、Ｚ値のビット１５～８をＬＵＴ１のインデックス番号に設定し、ＬＵＴ１を用いて仮想ポリゴンにテクスチャマッピングを行い、その仮想ポリゴンを別バッファに描画する（ステップＳ３）。

【０１７５】

次に、Ｚ値のビット２３～１６をＬＵＴ２のインデックス番号に設定し、ＬＵＴ２を用いて仮想ポリゴンにテクスチャマッピングを行い、その仮想ポリゴンを別バッファに描画する（ステップＳ４）。この際、図１７で説明したように、８ビット化されたＺ値の下位の４ビット（データ有効ビット）については上書きされないようにマスクしておく。

【０１７６】

次に、ステップＳ４で得られた８ビットのＺ２値をＬＵＴ３のインデックス番号に設定し、ＬＵＴ３を用いて仮想ポリゴンにテクスチャマッピングを行い、その仮想ポリゴンをフレームバッファ（αプレーン）に描画する（ステップＳ５）。

【０１７７】

次に、ステップＳ１でワークバッファに描画された元画像を、テクスチャ座標Ｕ、Ｖを（０．５、０．５）だけシフトしてバイリニアフィルタ方式で仮想ポリゴンにマッピングしながら、その仮想ポリゴンを別バッファに描画する（ステップＳ６）。

【０１７８】

次に、ステップＳ６で別バッファに描画された画像を、テクスチャ座標Ｕ、Ｖを（－０．５、－０．５）だけシフトしてバイリニアフィルタ方式で仮想ポリゴンにマッピングしながら、その仮想ポリゴンをフレームバッファに描画する（ステップＳ７）。この際、ステップＳ５でフレームバッファに描画されたα値を用いてαブレンディングを行い、元画像とぼかし画像のα合成を行う。

【０１７９】

以上のようにして、いわゆる被写界深度の表現が可能になる。

【０１８０】

４．ハードウェア構成

次に、本実施形態を実現できるハードウェアの構成の一例について図２９を用いて説明する。

【０１８１】

メインプロセッサ９００は、ＣＤ９８２（情報記憶媒体）に格納されたプログラム、通信インターフェース９９０を介して転送されたプログラム、或いはＲＯＭ９５０（情報記憶媒体の１つ）に格納されたプログラムなどに基づき動作し、ゲーム処理、画像処理、音処理などの種々の処理を実行する。

【０１８２】

コプロセッサ９０２は、メインプロセッサ９００の処理を補助するものであり、高速並列演算が可能な積和算器や除算器を有し、マトリクス演算（ベクトル演算）を高速に実行する。例えば、オブジェクトを移動させたり動作（モーション）させるための物理シミュレーションに、マトリクス演算などの処理が必要な場合には、メインプロセッサ９００上で動作するプログラムが、その処理をコプロセッサ９０２に指示（依頼）する。

【０１８３】

ジオメトリプロセッサ９０４は、座標変換、透視変換、光源計算、曲面生成などのジオメトリ処理を行うものであり、高速並列演算が可能な積和算器や除算器を有し、マトリクス演算（ベクトル演算）を高速に実行する。例えば、座標変換、透視変換、光源計算などの処理を行う場合には、メインプロセッサ９００で動

作するプログラムが、その処理をジオメトリプロセッサ９０４に指示する。

【０１８４】

データ伸張プロセッサ９０６は、圧縮された画像データや音データを伸張するデコード処理を行ったり、メインプロセッサ９００のデコード処理をアクセレートする処理を行う。これにより、オープニング画面、インターミッション画面、エンディング画面、或いはゲーム画面などにおいて、ＭＰＥＧ方式等で圧縮された動画像を表示できるようになる。なお、デコード処理の対象となる画像データや音データは、ＲＯＭ９５０、ＣＤ９８２に格納されたり、或いは通信インターフェース９９０を介して外部から転送される。

【０１８５】

描画プロセッサ９１０は、ポリゴンや曲面などのプリミティブ面で構成されるオブジェクトの描画（レンダリング）処理を高速に実行するものである。オブジェクトの描画の際には、メインプロセッサ９００は、ＤＭＡコントローラ９７０の機能を利用して、オブジェクトデータを描画プロセッサ９１０に渡すと共に、必要であればテクスチャ記憶部９２４にテクスチャを転送する。すると、描画プロセッサ９１０は、これらのオブジェクトデータやテクスチャに基づいて、Ｚバッファなどを利用した陰面消去を行いながら、オブジェクトをフレームバッファ９２２に高速に描画する。また、描画プロセッサ９１０は、αブレンディング（半透明処理）、デプスキューイング、ミップマッピング、フォグ処理、バイリニア・フィルタリング、トライリニア・フィルタリング、アンチエリアシング、シェーディング処理なども行うことができる。そして、１フレーム分の画像がフレームバッファ９２２に書き込まれると、その画像はディスプレイ９１２に表示される。

【０１８６】

サウンドプロセッサ９３０は、多チャンネルのＡＤＰＣＭ音源などを内蔵し、ＢＧＭ、効果音、音声などの高品位のゲーム音を生成する。生成されたゲーム音は、スピーカ９３２から出力される。

【０１８７】

ゲームコントローラ９４２からの操作データや、メモリカード９４４からのセ

ーブデータ、個人データは、シリアルインターフェース９４０を介してデータ転送される。

【０１８８】

ＲＯＭ９５０にはシステムプログラムなどが格納される。なお、業務用ゲームシステムの場合には、ＲＯＭ９５０が情報記憶媒体として機能し、ＲＯＭ９５０に各種プログラムが格納されることになる。なお、ＲＯＭ９５０の代わりにハードディスクを利用するようにしてもよい。

【０１８９】

ＲＡＭ９６０は、各種プロセッサの作業領域として用いられる。

【０１９０】

ＤＭＡコントローラ９７０は、プロセッサ、メモリ（ＲＡＭ、ＶＲＡＭ、ＲＯＭ等）間でのＤＭＡ転送を制御するものである。

【０１９１】

ＣＤドライブ９８０は、プログラム、画像データ、或いは音データなどが格納されるＣＤ９８２（情報記憶媒体）を駆動し、これらのプログラム、データへのアクセスを可能にする。

【０１９２】

通信インターフェース９９０は、ネットワークを介して外部との間でデータ転送を行うためのインターフェースである。この場合に、通信インターフェース９９０に接続されるネットワークとしては、通信回線（アナログ電話回線、ＩＳＤＮ）、高速シリアルバスなどを考えることができる。そして、通信回線を利用することでインターネットを介したデータ転送が可能になる。また、高速シリアルバスを利用することで、他のゲームシステムとの間でのデータ転送が可能になる。

【０１９３】

なお、本発明の各手段は、その全てを、ハードウェアのみにより実行してもよいし、情報記憶媒体に格納されるプログラムや通信インターフェースを介して配信されるプログラムのみにより実行してもよい。或いは、ハードウェアとプログラムの両方により実行してもよい。

【０１９４】

そして、本発明の各手段をハードウェアとプログラムの両方により実行する場合には、情報記憶媒体には、本発明の各手段をハードウェアを利用して実行するためのプログラムが格納されることになる。より具体的には、上記プログラムが、ハードウェアである各プロセッサ９０２、９０４、９０６、９１０、９３０等に処理を指示すると共に、必要であればデータを渡す。そして、各プロセッサ９０２、９０４、９０６、９１０、９３０等は、その指示と渡されたデータとに基づいて、本発明の各手段を実行することになる。

【０１９５】

図３０（Ａ）に、本実施形態を業務用ゲームシステムに適用した場合の例を示す。プレーヤは、ディスプレイ１１００上に映し出されたゲーム画像を見ながら、レバー１１０２、ボタン１１０４等を操作してゲームを楽しむ。内蔵されるシステムボード（サーキットボード）１１０６には、各種プロセッサ、各種メモリなどが実装される。そして、本発明の各手段を実行するための情報（プログラム又はデータ）は、システムボード１１０６上の情報記憶媒体であるメモリ１１０８に格納される。以下、この情報を格納情報と呼ぶ。

【０１９６】

図３０（Ｂ）に、本実施形態を家庭用のゲームシステムに適用した場合の例を示す。プレーヤはディスプレイ１２００に映し出されたゲーム画像を見ながら、ゲームコントローラ１２０２、１２０４を操作してゲームを楽しむ。この場合、上記格納情報は、本体システムに着脱自在な情報記憶媒体であるＣＤ１２０６、或いはメモリカード１２０８、１２０９等に格納されている。

【０１９７】

図３０（Ｃ）に、ホスト装置１３００と、このホスト装置１３００とネットワーク１３０２（ＬＡＮのような小規模ネットワークや、インターネットのような広域ネットワーク）を介して接続される端末１３０４-1～１３０４-n（ゲーム装置、携帯型電話）とを含むシステムに本実施形態を適用した場合の例を示す。この場合、上記格納情報は、例えばホスト装置１３００が制御可能な磁気ディスク装置、磁気テープ装置、メモリ等の情報記憶媒体１３０６に格納されている。端

末１３０４-1～１３０４-nが、スタンドアロンでゲーム画像、ゲーム音を生成できるものである場合には、ホスト装置１３００からは、ゲーム画像、ゲーム音を生成するためのゲームプログラム等が端末１３０４-1～１３０４-nに配送される。一方、スタンドアロンで生成できない場合には、ホスト装置１３００がゲーム画像、ゲーム音を生成し、これを端末１３０４-1～１３０４-nに伝送し端末において出力することになる。

【０１９８】

なお、図３０（Ｃ）の構成の場合に、本発明の各手段を、ホスト装置（サーバー）と端末とで分散して実行するようにしてもよい。また、本発明の各手段を実行するための上記格納情報を、ホスト装置（サーバー）の情報記憶媒体と端末の情報記憶媒体に分散して格納するようにしてもよい。

【０１９９】

またネットワークに接続する端末は、家庭用ゲームシステムであってもよいし業務用ゲームシステムであってもよい。そして、業務用ゲームシステムをネットワークに接続する場合には、業務用ゲームシステムとの間で情報のやり取りが可能であると共に家庭用ゲームシステムとの間でも情報のやり取りが可能な携帯型情報記憶装置（メモリカード、携帯型ゲーム装置）を用いることが望ましい。

【０２００】

なお本発明は、上記実施形態で説明したものに限らず、種々の変形実施が可能である。

【０２０１】

例えば、本発明のうち従属請求項に係る発明においては、従属先の請求項の構成要件の一部を省略する構成とすることもできる。また、本発明の１の独立請求項に係る発明の要部を、他の独立請求項に従属させることもできる。

【０２０２】

また、奥行き値の第２の奥行き値への変換は、図１７で説明したようにインデックスカラー・テクスチャマッピング用のルックアップテーブルを利用する手法で実現することが特に望ましいが、他の手法により実現してもよい。

【０２０３】

またルックアップテーブルの変換特性も図１８、図１９、図２０（Ａ）、（Ｂ）に示した変換特性に限定されず、種々の変形実施が可能である。

【０２０４】

また、元画像の合成対象となるぼかし画像は、図２３、図２４で説明した手法により生成することが特に望ましいが、これに限定されない。例えば元画像と元画像をずらした画像を合成したり、当該フレームの元画像と前のフレームの元画像とを合成したりしてぼかし画像を生成してもよい。

【０２０５】

また、テクセル補間方式を利用してぼかし画像を生成する発明も、図２３～図２６（Ｂ）で説明した手法に限定されない。例えば、画面全体をぼかすのではなく、画面よりも小さいぼかし領域を設定して、その領域にある元画像をぼかすようにしてもよい。

【０２０６】

また本実施形態では視点から近いほど奥行き値が大きくなる場合を例にとり説明したが、視点から遠いほど奥行き値が大きくなる場合にも本発明は適用できる。

【０２０７】

また、本発明は種々のゲーム（格闘ゲーム、シューティングゲーム、ロボット対戦ゲーム、スポーツゲーム、競争ゲーム、ロールプレイングゲーム、音楽演奏ゲーム、ダンスゲーム等）に適用できる。

【０２０８】

また本発明は、業務用ゲームシステム、家庭用ゲームシステム、多数のプレーヤが参加する大型アトラクションシステム、シミュレータ、マルチメディア端末、ゲーム画像を生成するシステムボード等の種々のゲームシステム（画像生成システム）に適用できる。

【図面の簡単な説明】

【図１】

本実施形態のゲームシステムのブロック図の例である。

【図２】

図２（Ａ）、（Ｂ）は、ガンマ補正を実現する第１、第２の手法について説明するための図である。

【図３】

インデックスカラー・テクスチャマッピングについて説明するための図である。

【図４】

インデックスカラー・テクスチャマッピング用のＬＵＴを有効利用して、元画像を変換する手法について説明するための図である。

【図５】

図５（Ａ）、（Ｂ）は、本実施形態により生成されるゲーム画像の例である。

【図６】

元画像を複数のブロックに分割し、各ブロックの画像を、ＬＵＴを用いて分割ブロックサイズのポリゴンにテクスチャマッピングする手法について説明するための図である。

【図７】

ＬＵＴの変換により得られた画像情報のうち、必要な画像情報のみが描画領域に描画され、他の画像情報が描画されないようにマスク処理を行う手法について説明するための図である。

【図８】

ＬＵＴを利用したテクスチャマッピングを行い、αプレーンを作成する手法について説明するための図である。

【図９】

Ｚ値をＬＵＴのインデックス番号に設定する手法について説明するための図である。

【図１０】

Ｚ値に応じたα値を設定し、設定されたα値を用いて元画像とぼかし画像を合成する手法について説明するための図である。

【図１１】

図１１（Ａ）、（Ｂ）は、元画像とそのぼかし画像の例である。

【図１２】

Ｚ値に応じたα値の設定手法について説明するための図である。

【図１３】

図１３（Ａ）、（Ｂ）、（Ｃ）は、仮想オブジェクトを描画することで仮想オブジェクトの奥側の画素のα値を更新する手法について説明するための図である。

【図１４】

Ｚ値をＺ２値に変換し、そのＺ２値をα値に変換して元画像とぼかし画像を合成する手法について説明するための図である。

【図１５】

Ｚ値の最上位ビットを含む上位ビットでＺ２値を構成した場合の問題点について説明するための図である。

【図１６】

Ｚ値の最上位ビットよりも下位のビットＩ～Ｊビットでの２値を構成すると共にＺ２値を所与の値にクランプする手法について説明するための図である。

【図１７】

ＬＵＴを用いてＺ値をＺ２値に変換する手法について説明するための図である。

【図１８】

Ｚ値のビット１５～８を変換するＬＵＴ１の具体例を示す図である。

【図１９】

Ｚ値のビット２３～１６を変換するＬＵＴ２の具体例を示す図である。

【図２０】

図２０（Ａ）、（Ｂ）は、Ｚ２値をα値に変換するＬＵＴ３の具体例とその変換特性曲線の例を示す図である。

【図２１】

クランプ処理について説明するための図である。

【図２２】

バイリニアフィルタ方式のテクスチャマッピングについて説明するための図で

ある。

【図２３】

バイリニアフィルタ方式を有効利用してぼかし画像を生成する手法について説明するための図である。

【図２４】

バイリニアフィルタ方式を有効利用してぼかし画像を生成する手法について説明するための図である。

【図２５】

図２５（Ａ）、（Ｂ）は、バイリニアフィルタ方式の補間機能によりぼかし画像が生成される原理について説明するための図である。

【図２６】

図２６（Ａ）、（Ｂ）も、バイリニアフィルタ方式の補間機能によりぼかし画像が生成される原理について説明するための図である。

【図２７】

本実施形態の処理の詳細例について示すフローチャートである。

【図２８】

本実施形態の処理の詳細例について示すフローチャートである。

【図２９】

本実施形態を実現できるハードウェアの構成の一例を示す図である。

【図３０】

図３０（Ａ）、（Ｂ）、（Ｃ）は、本実施形態が適用される種々の形態のシステムの例を示す図である。

【符号の説明】

１０　仮想カメラ

１００　処理部

１１０　ゲーム処理部

１１２　移動・動作演算部

１３０　画像生成部

１３２　ジオメトリ処理部

１３４　インデックス番号設定部

１４０　描画部

１４２　テクスチャマッピング部

１４４　α合成部

１４６　陰面消去部

１５０　音生成部

１６０　操作部

１７０　記憶部

１７２　主記憶領域

１７４　フレームバッファ

１７６　テクスチャ記憶部

１７８　ＬＵＴ記憶部

１７９　Ｚバッファ

１８０　情報記憶媒体

１９０　表示部

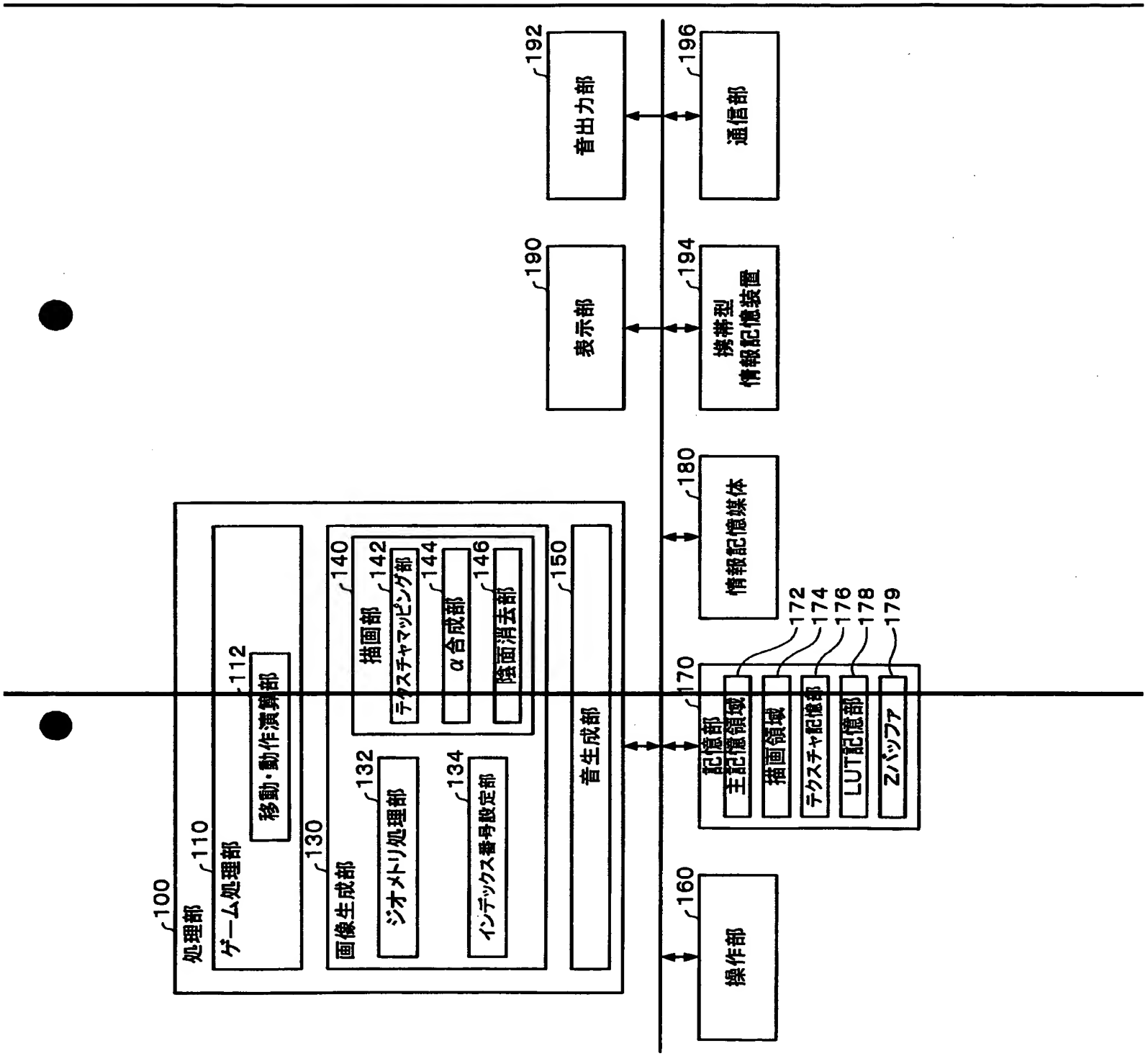
１９２　音出力部

１９４　携帯型情報記憶装置

１９６　通信部

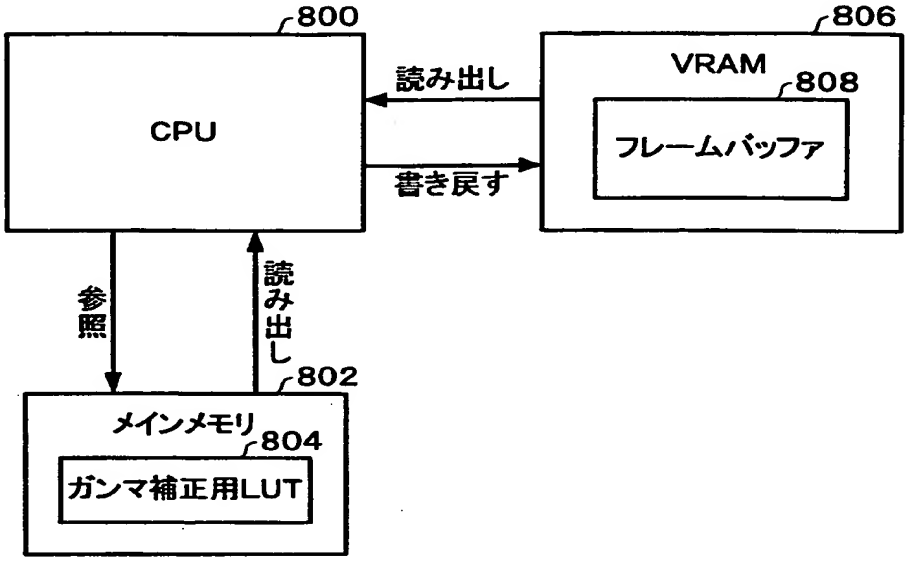
【書類名】　　図面

【図１】

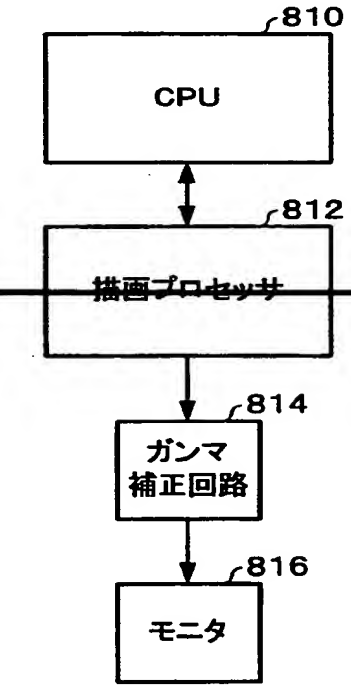

100
処理部
110
ゲーム処理部
112
移動・動作演算部
130
画像生成部
132
ジオメトリ処理部
134
インデックス番号設定部
140
描画部
142
テクスチャマッピング部
144
α合成部
146
陰面消去部
150
音生成部
160
操作部
170
記憶部
172
主記憶領域
174
描画領域
176
テクスチャ記憶部
178
LUT記憶部
179
Zバッファ
180
情報記憶媒体
190
表示部
192
音出力部
194
携帯型情報記憶装置
196
通信部

【図２】

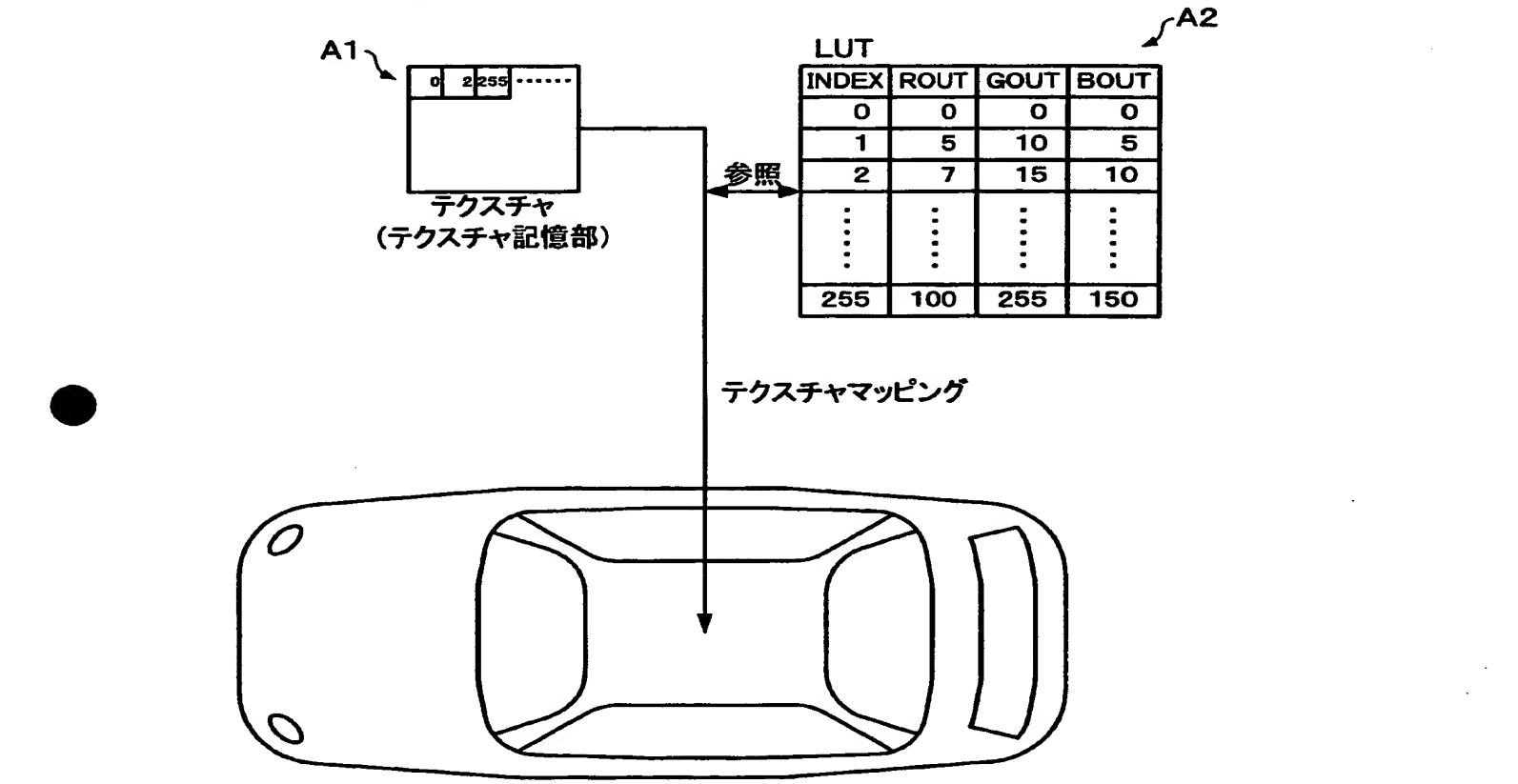
(A)


800
CPU
806
VRAM
808
フレームバッファ
読み出し
書き戻す
参照
読み出し
802
メインメモリ
804
ガンマ補正用LUT

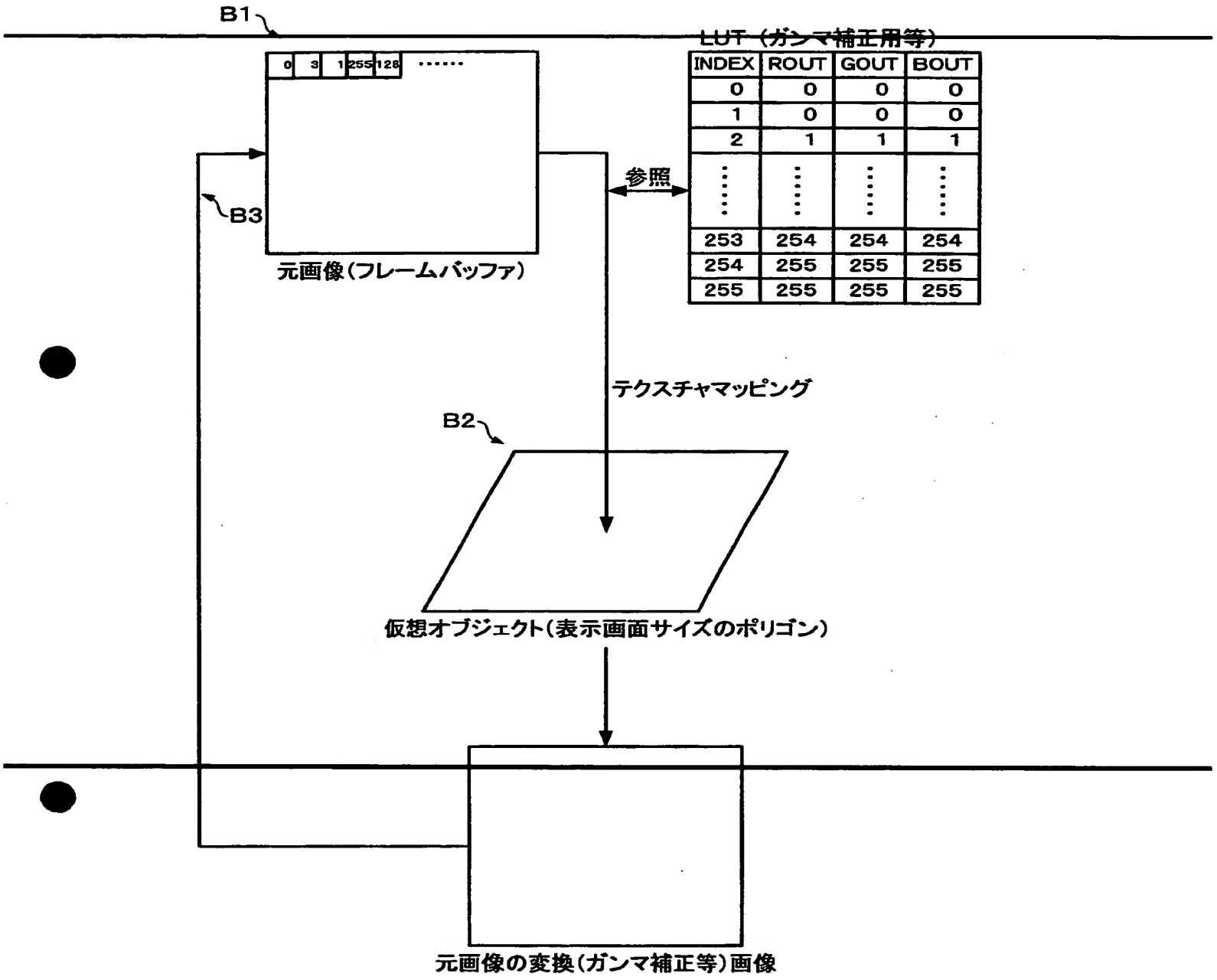
(B)


810
CPU
812
描画プロセッサ
814
ガンマ
補正回路
816
モニタ

【図３】

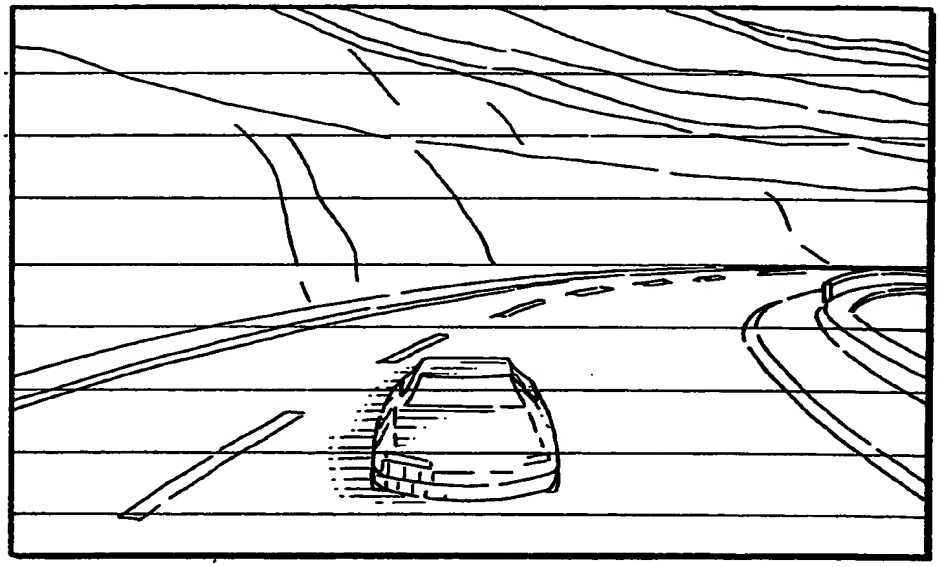

インデックスカラー・テクスチャマッピング
A1
0
2
255
テクスチャ
（テクスチャ記憶部）
参照
A2
LUT
INDEX
ROUT
GOUT
BOUT
0 0 0 0
1 5 10 5
2 7 15 10
255 100 255 150
テクスチャマッピング

【図４】

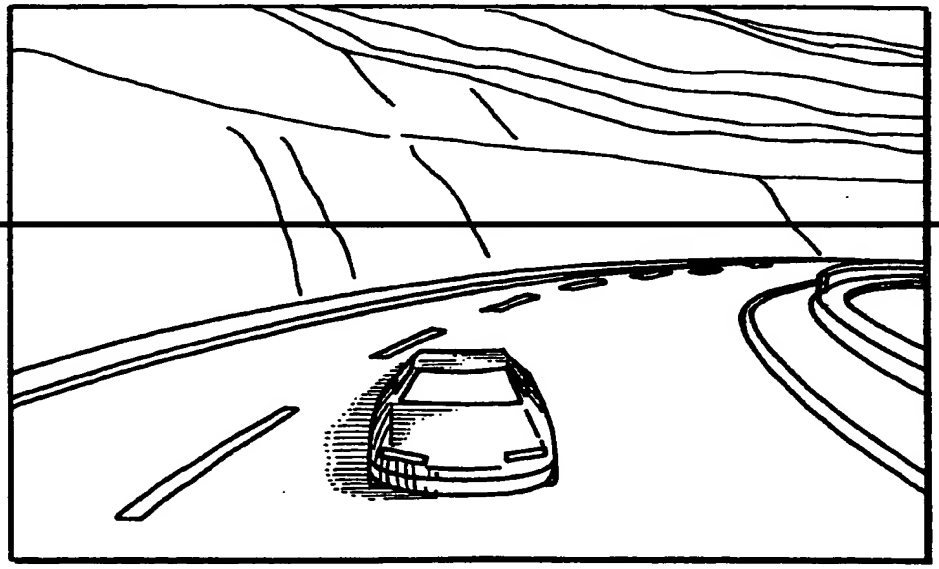

B1
LUT（ガンマ補正用等）
INDEX ROUT GOUT BOUT
0 0 0 0
1 0 0 0
2 1 1 1
253 254 254 254
254 255 255 255
255 255 255 255
0 3 1 255 128 ……
参照
B3
元画像（フレームバッファ）
テクスチャマッピング
B2
仮想オブジェクト（表示画面サイズのポリゴン）
元画像の変換（ガンマ補正等）画像

【図５】

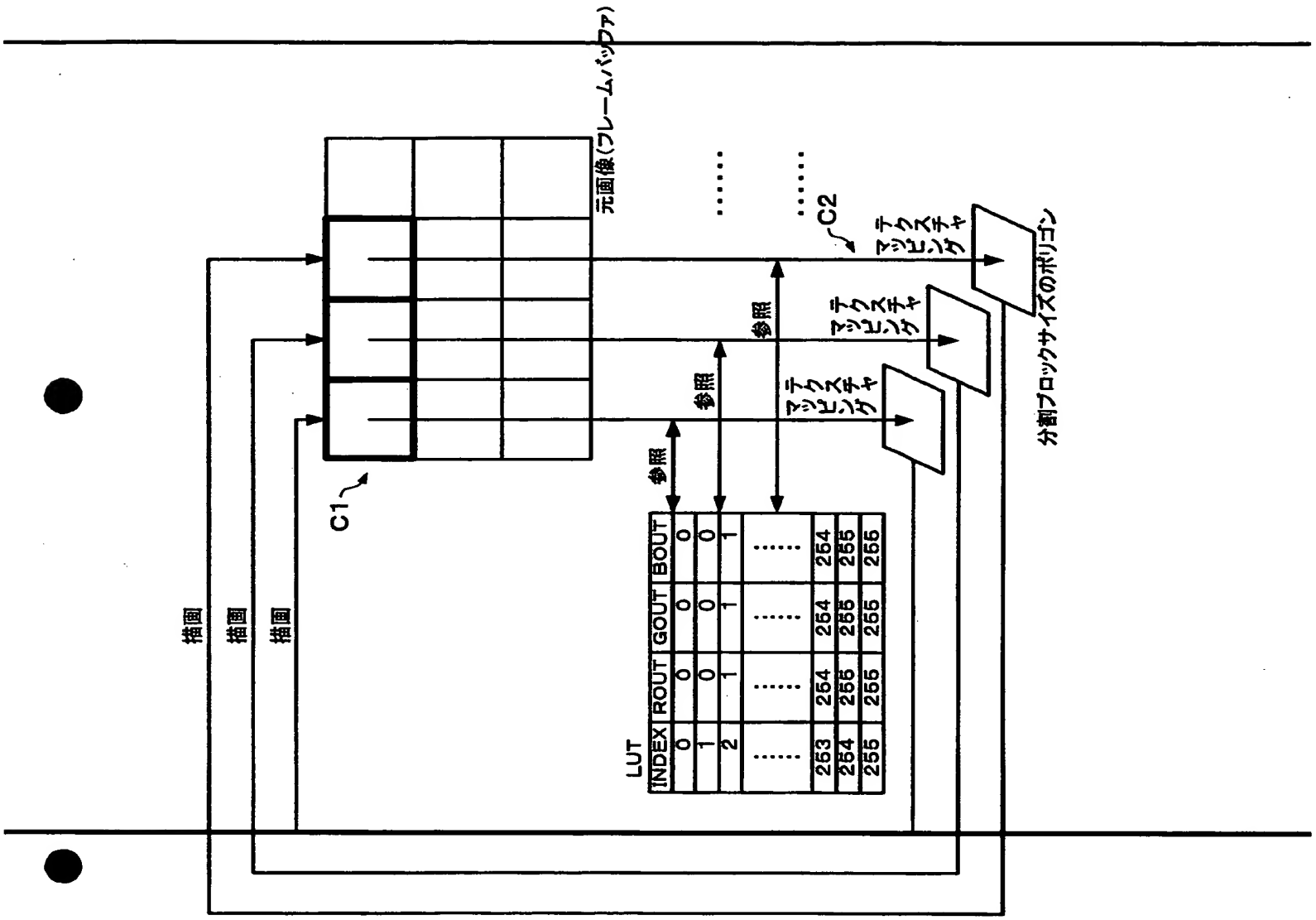
(A)



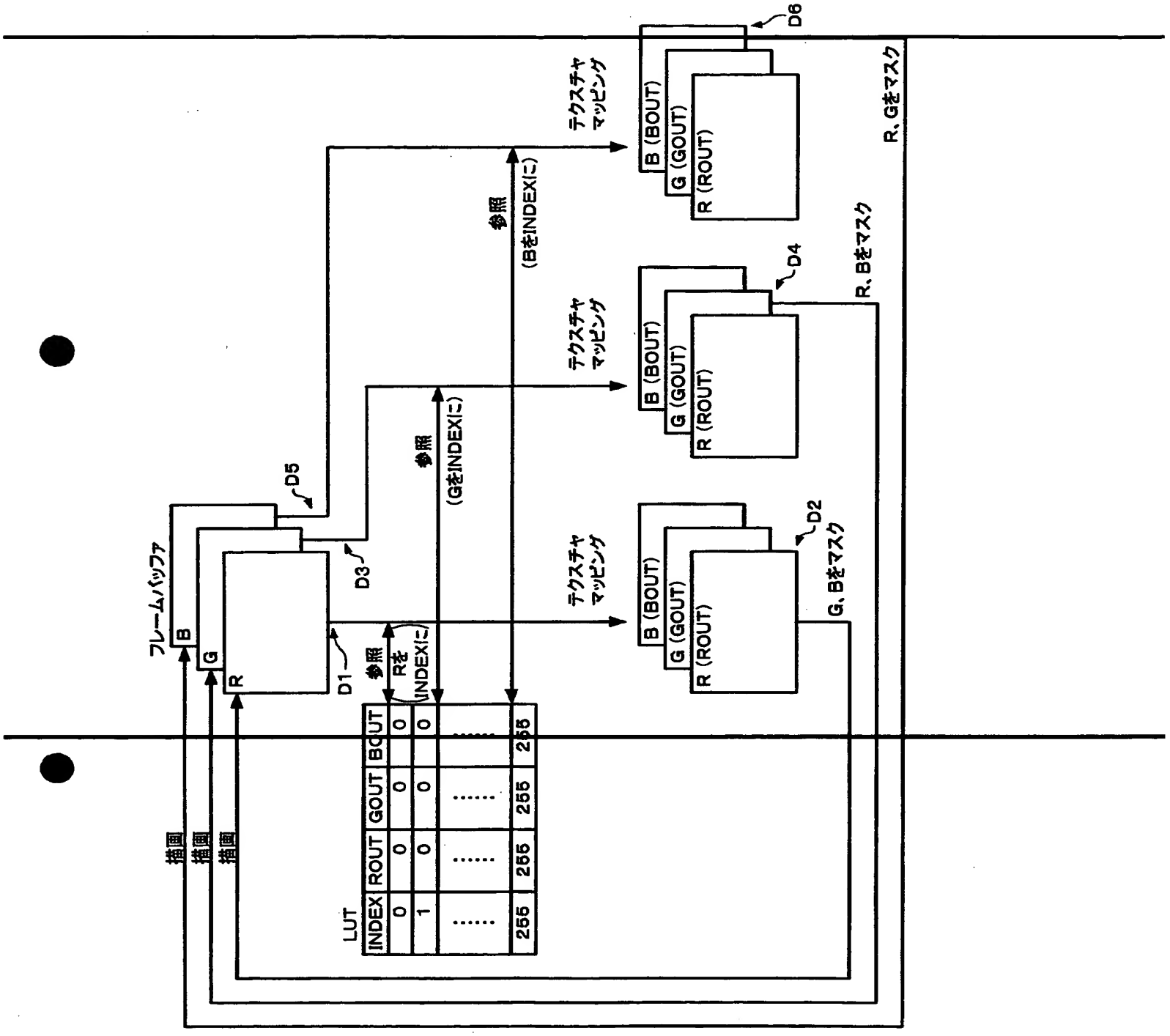
(B)



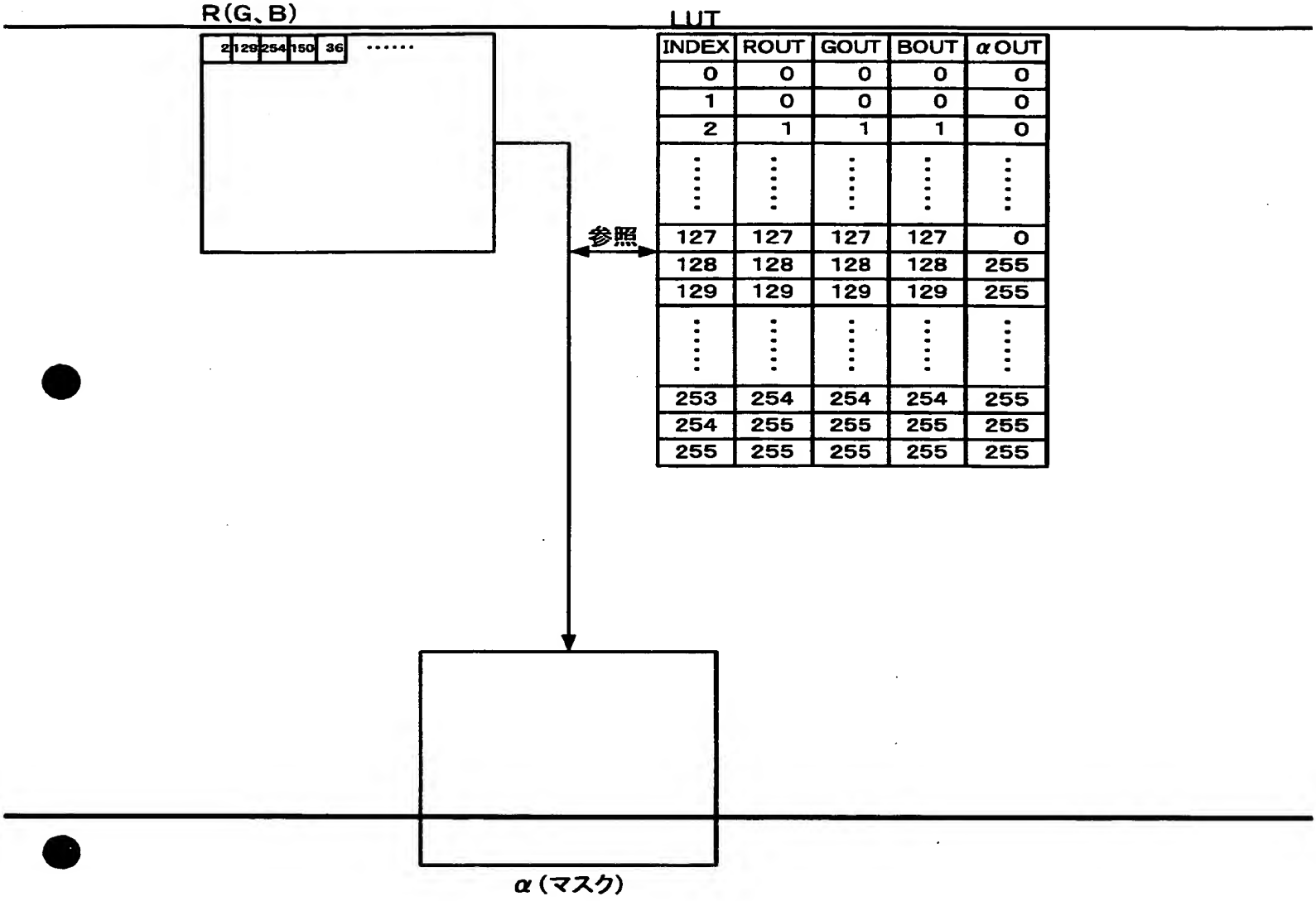
【図６】


元画像(フレームバッファ)
C1
C2
描画
描画
描画
参照
参照
参照
テクスチャマッピング
テクスチャマッピング
テクスチャマッピング
分割ブロックサイズのポリゴン
LUT
INDEX
ROUT
GOUT
BOUT

【図７】

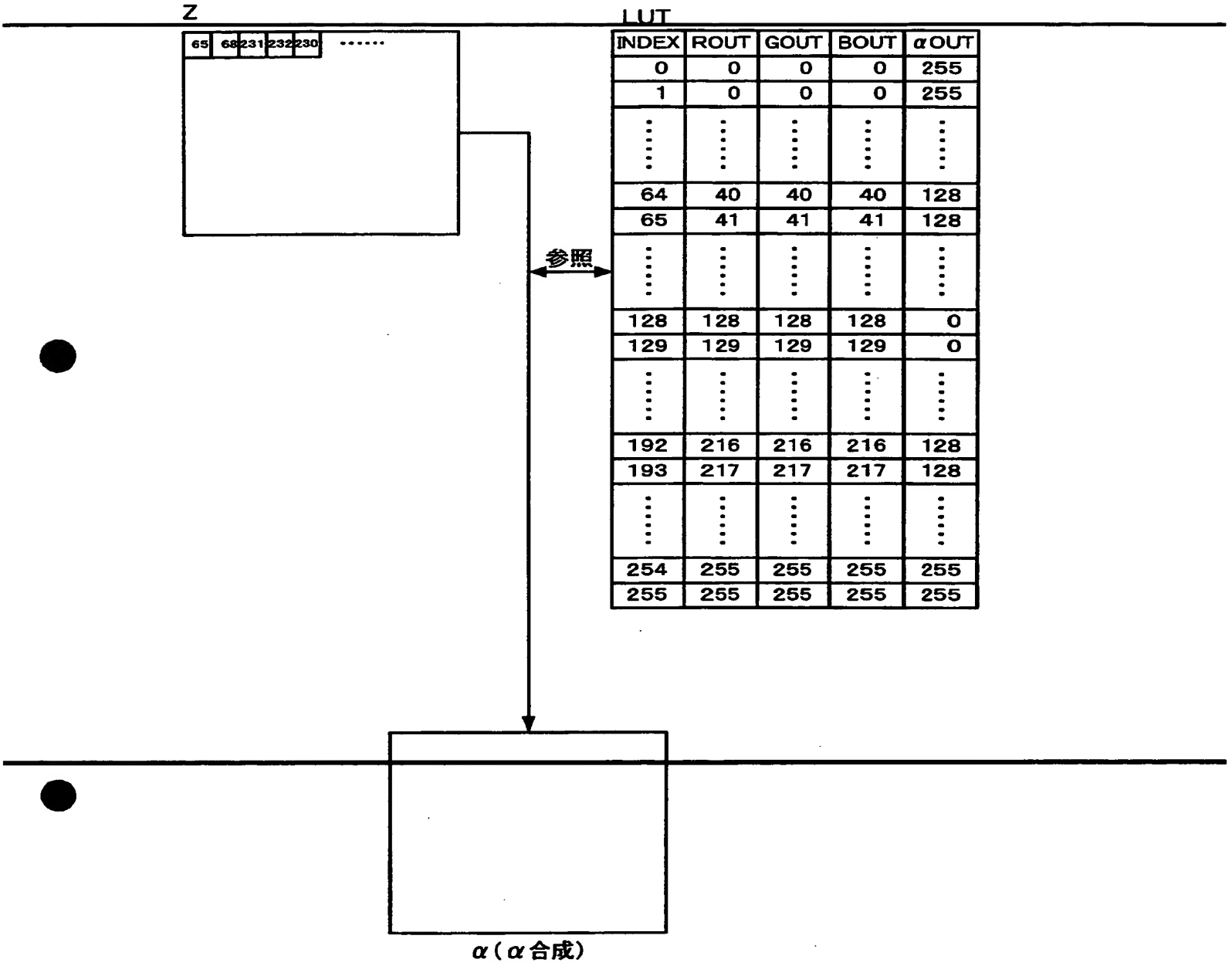

D6
テクスチャマッピング
B (BOUT)
G (GOUT)
R (ROUT)
R、Gをマスク
参照
(BをINDEXに)
D4
R、Bをマスク
D5
D3
(GをINDEXに)
D2
G、Bをマスク
フレームバッファ
B
G
R
D1
参照
(RをINDEXに)
描画
LUT
INDEX
ROUT
GOUT
BOUT
255

【図８】



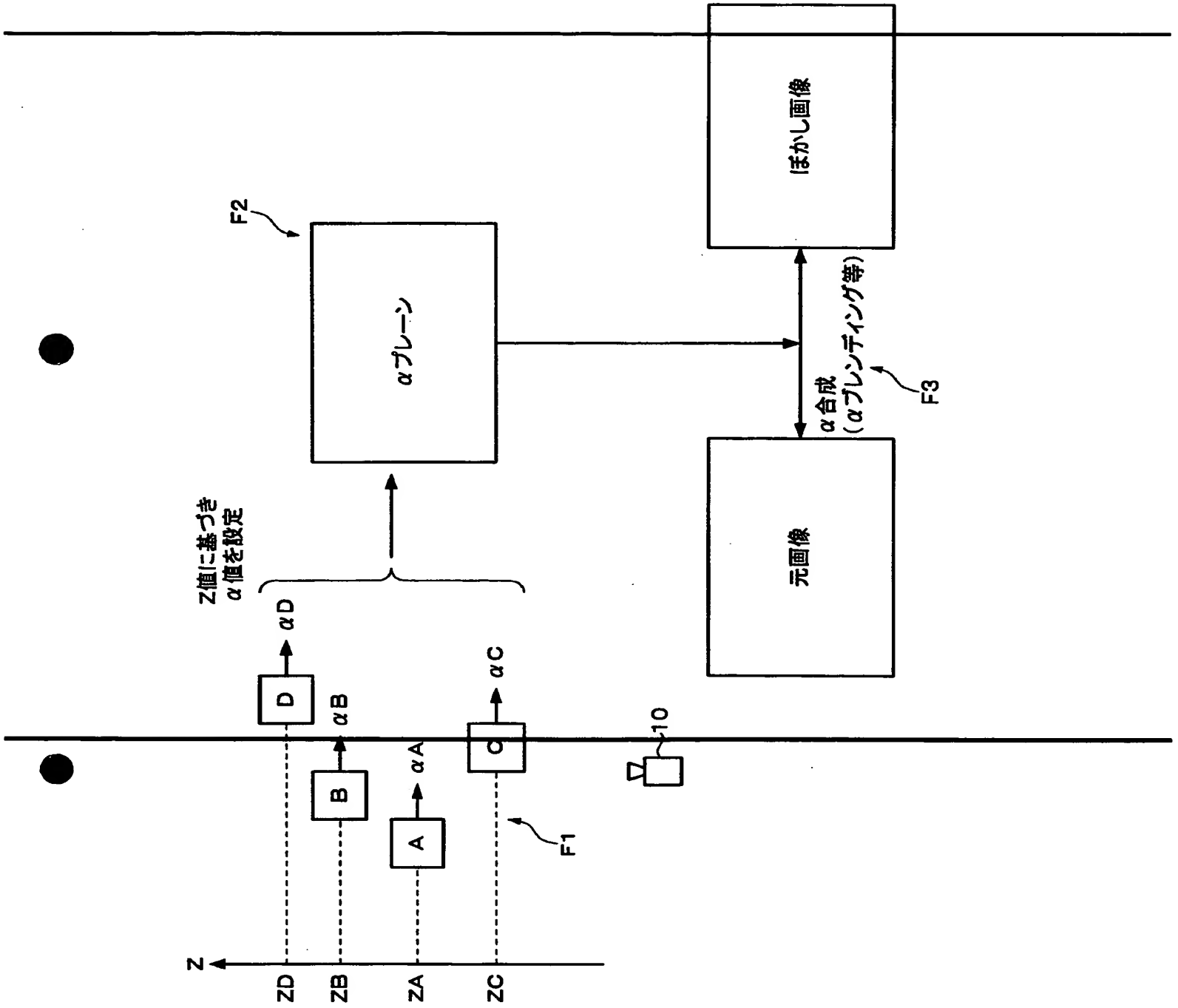
INDEX	ROUT	GOUT	BOUT	αOUT
0	0	0	0	0
1	0	0	0	0
2	1	1	1	0
⋮	⋮	⋮	⋮	⋮
127	127	127	127	0
128	128	128	128	255
129	129	129	129	255
⋮	⋮	⋮	⋮	⋮
253	254	254	254	255
254	255	255	255	255
255	255	255	255	255

【図９】



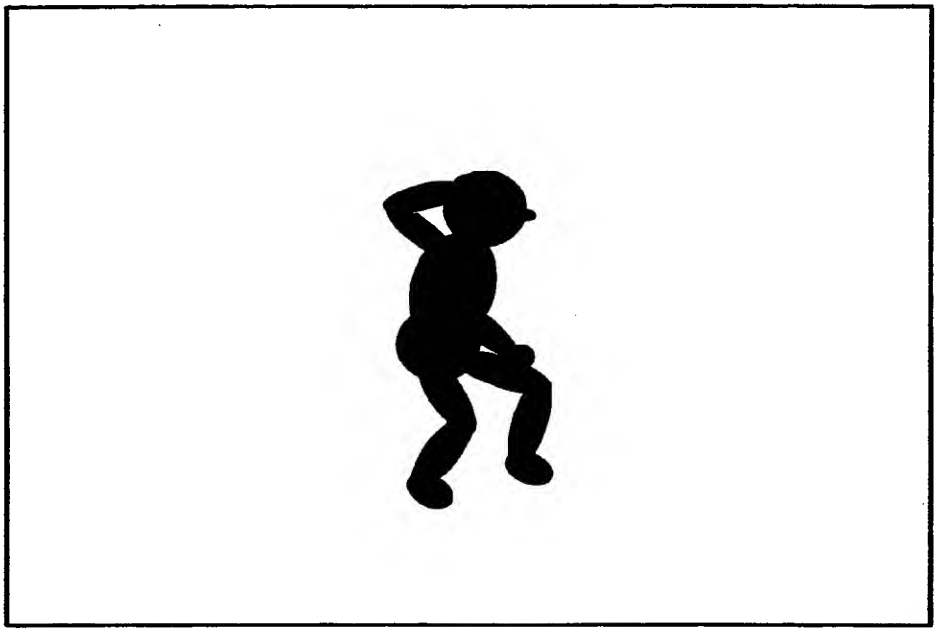
INDEX	ROUT	GOUT	BOUT	αOUT
0	0	0	0	255
1	0	0	0	255
⋮	⋮	⋮	⋮	⋮
64	40	40	40	128
65	41	41	41	128
⋮	⋮	⋮	⋮	⋮
128	128	128	128	0
129	129	129	129	0
⋮	⋮	⋮	⋮	⋮
192	216	216	216	128
193	217	217	217	128
⋮	⋮	⋮	⋮	⋮
254	255	255	255	255
255	255	255	255	255

【図１０】

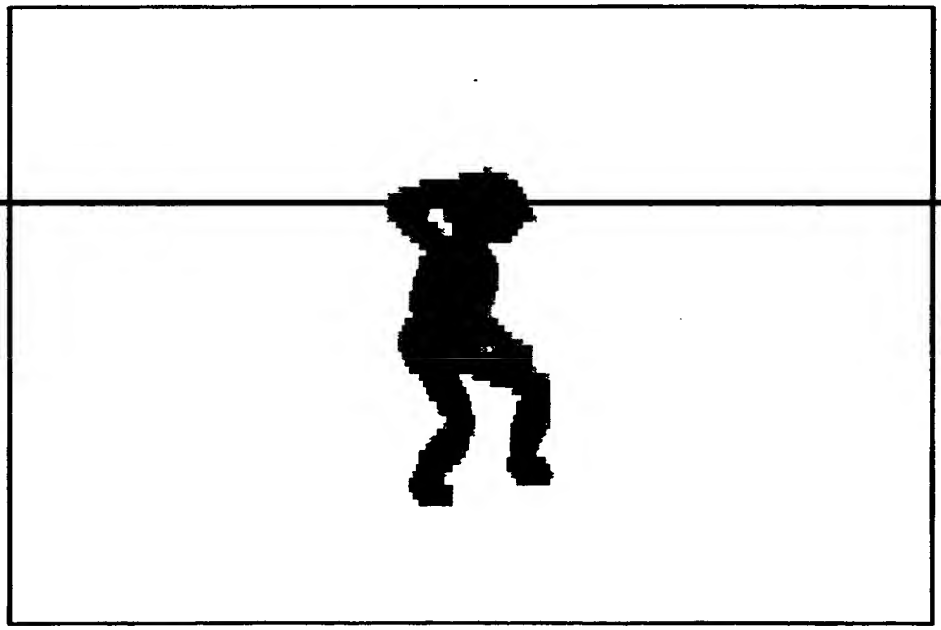

ぼかし画像
F2
αプレーン
α合成
(αブレンディング等)
F3
元画像
Z値に基づき
α値を設定
αD
D
αB
B
αA
A
αC
C
10
F1
Z
ZD
ZB
ZA
ZC

【図１１】

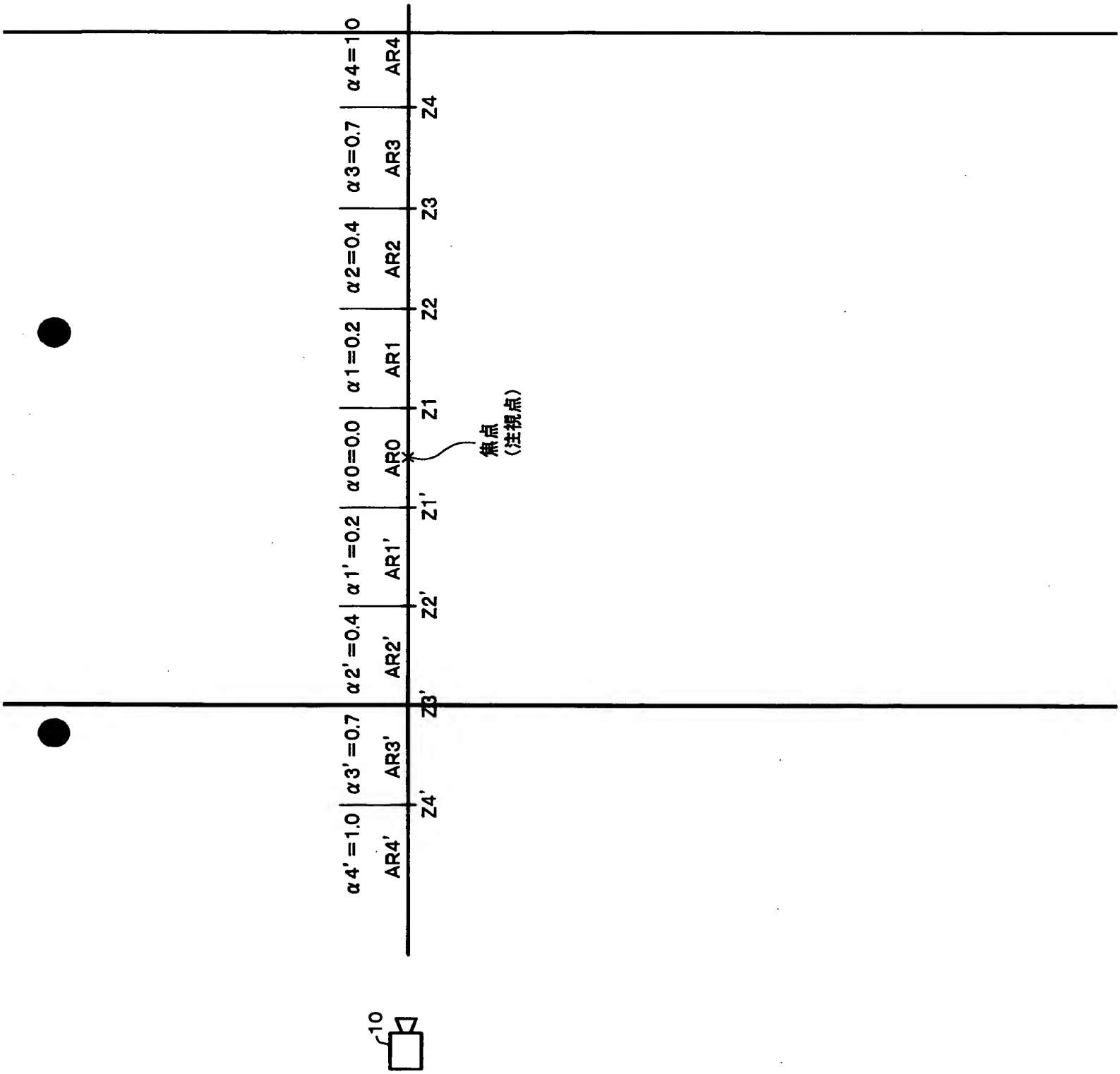
(A) 元画像



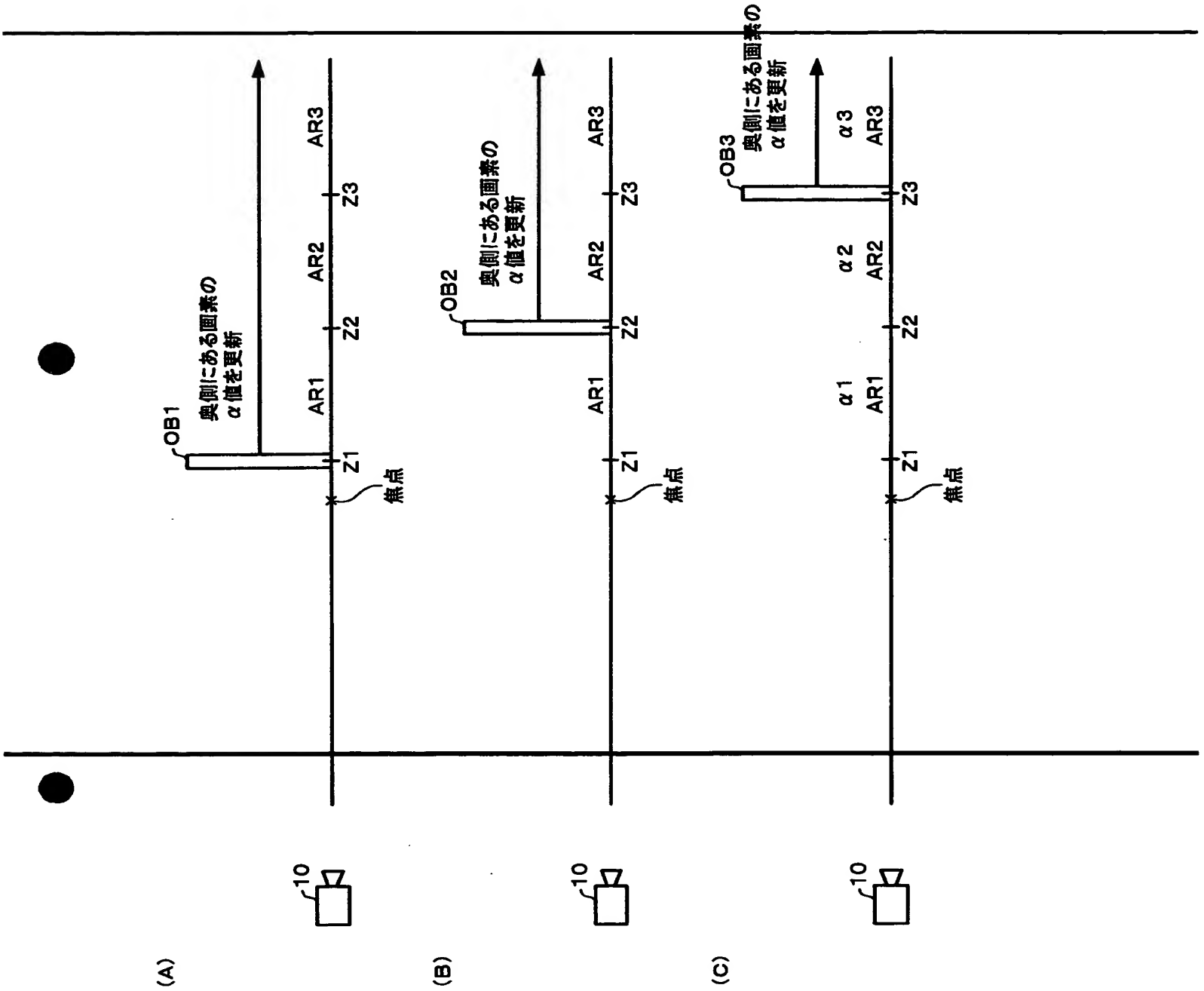
(B) ぼかし画像



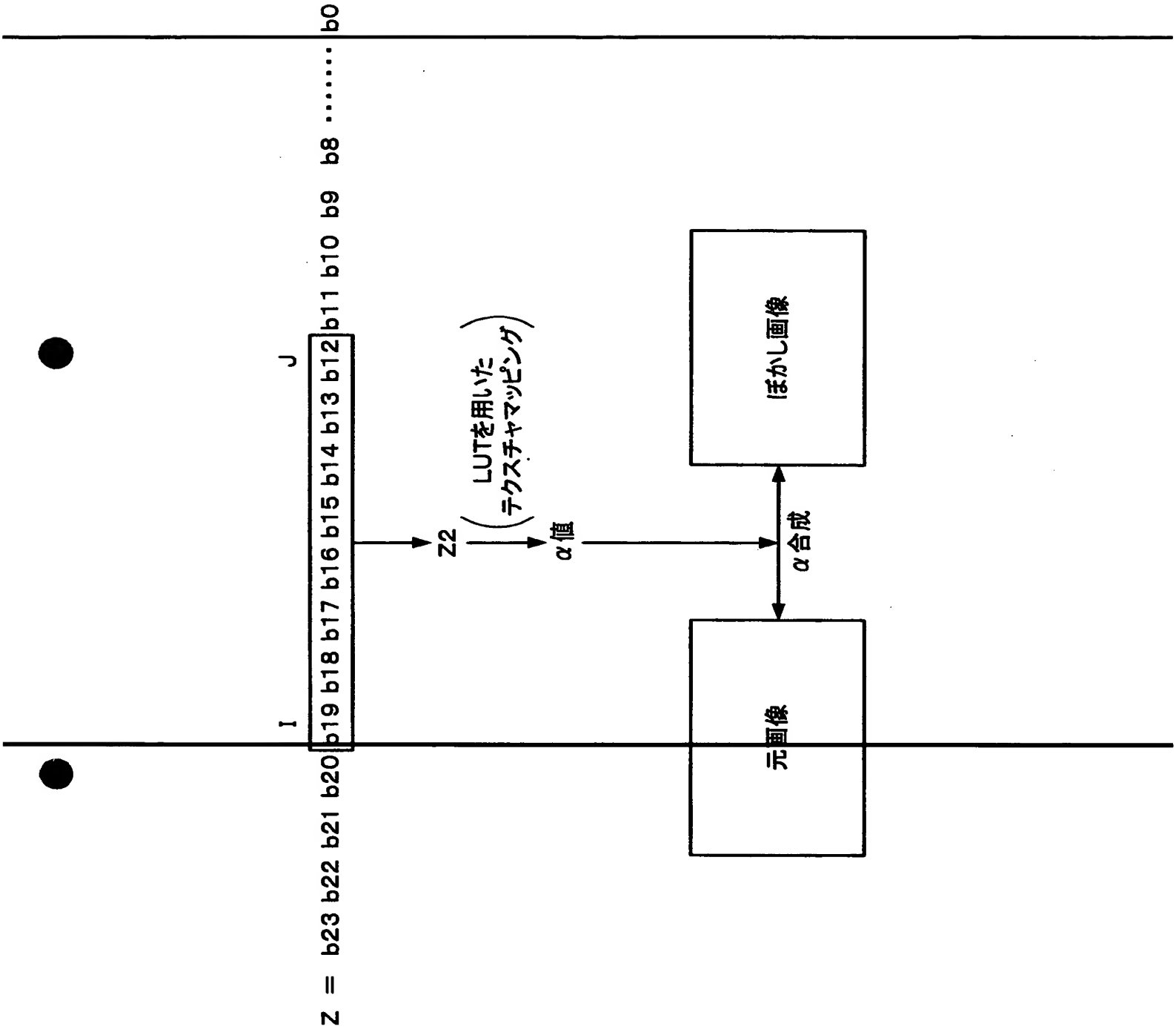
【図１２】


α4=1.0
AR4
Z4
α3=0.7
AR3
Z3
α2=0.4
AR2
Z2
α1=0.2
AR1
Z1
α0=0.0
AR0
Z1'
α1'=0.2
AR1'
Z2'
α2'=0.4
AR2'
Z3'
α3'=0.7
AR3'
Z4'
α4'=1.0
AR4'
無点
(注視点)
10

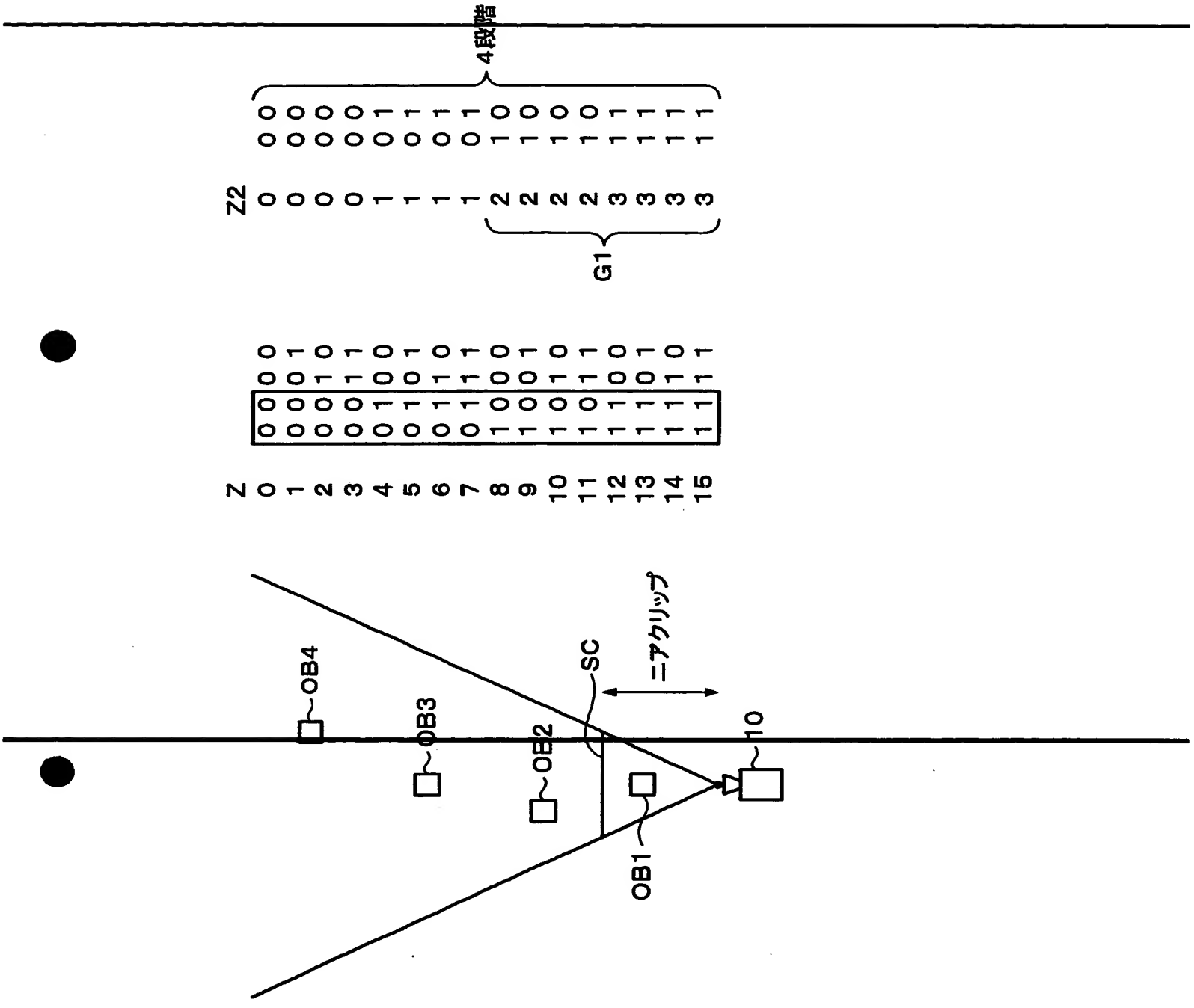
【図１３】


OB1
奥側にある画素の
α値を更新
AR1
AR2
AR3
Z1
Z2
Z3
焦点
10
(A)
OB2
奥側にある画素の
α値を更新
AR1
AR2
AR3
Z1
Z2
Z3
焦点
10
(B)
OB3
奥側にある画素の
α値を更新
α1
α2
α3
AR1
AR2
AR3
Z1
Z2
Z3
焦点
10
(C)

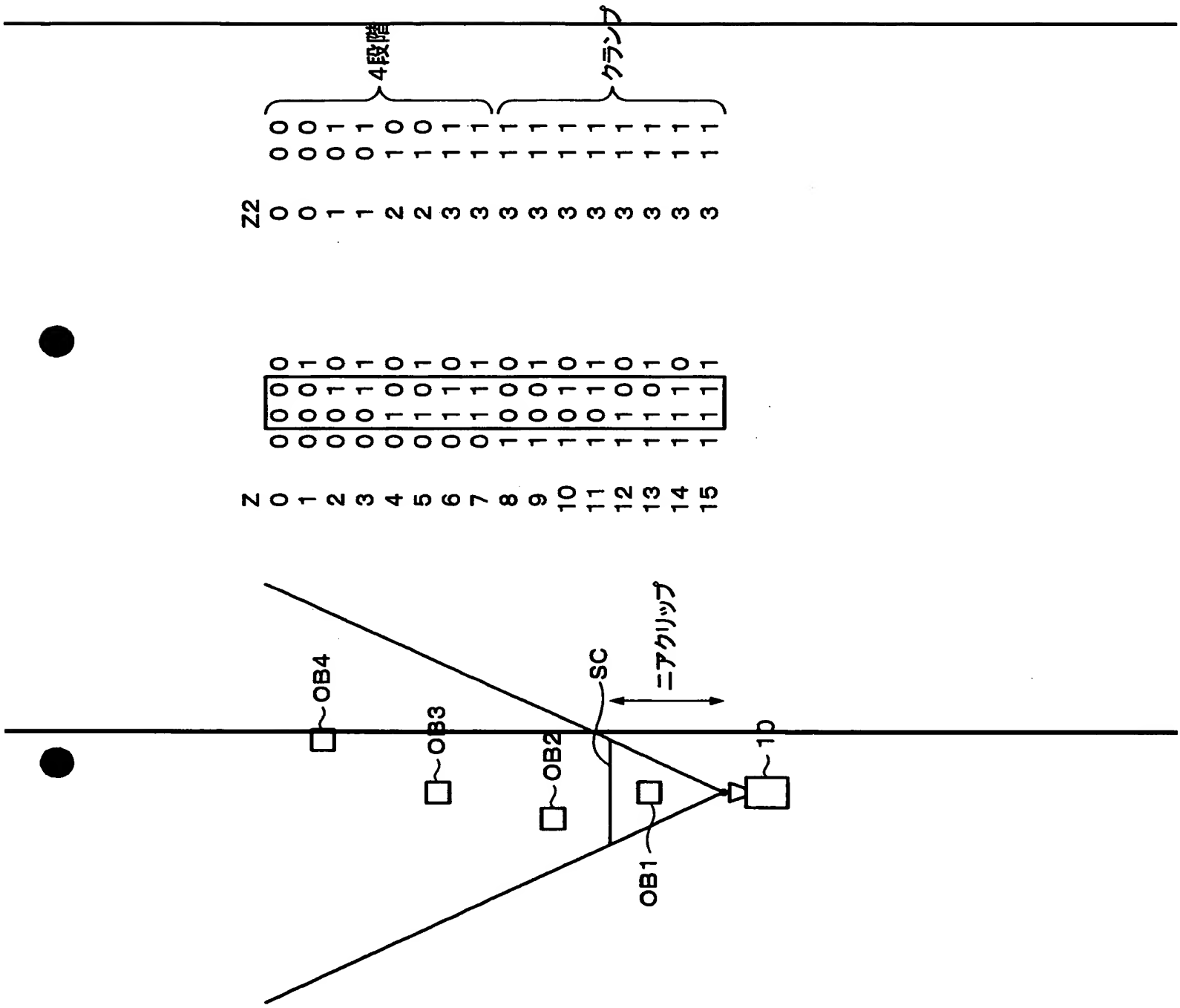
【図１４】


Z = b23 b22 b21 b20 b19 b18 b17 b16 b15 b14 b13 b12 b11 b10 b9 b8 ..... b0
I
J
Z2
LUTを用いた
テクスチャマッピング
α値
ぼかし画像
α合成
元画像

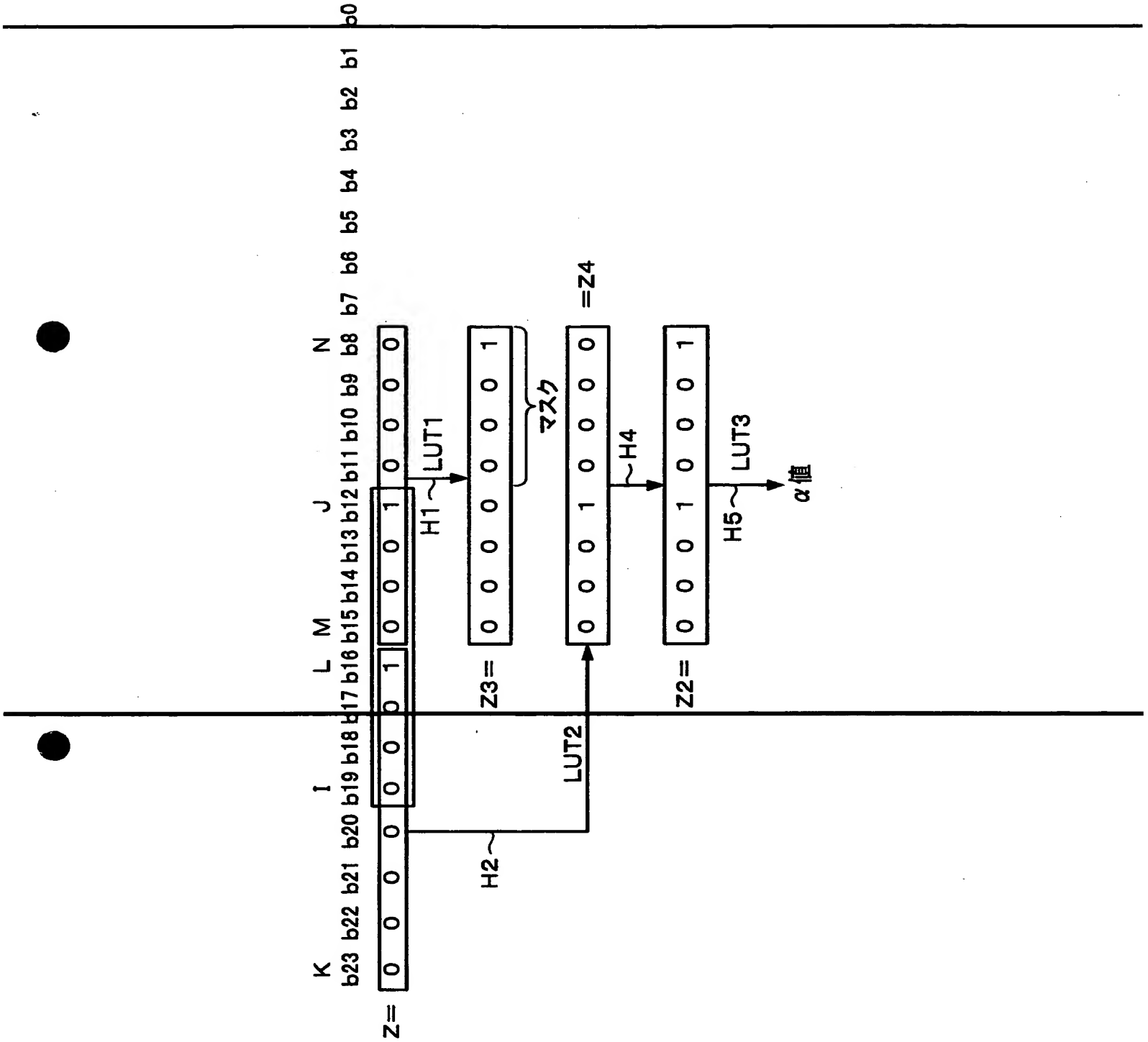
【図１５】



【図１６】


4段階
クランプ
Z2
0 0 1 1 2 2 3 3 3 3 3 3 3 3 3 3
Z
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
ニアクリップ
SC
OB4
OB3
OB2
OB1
1b

【図１７】


b0
b1
b2
b3
b4
b5
b6
b7
b8
b9
b10
b11
b12
b13
b14
b15
b16
b17
b18
b19
b20
b21
b22
b23
N
J
M
L
I
K
Z=
LUT1
H1
Z3=
マスク
=Z4
H4
LUT3
H5
α値
Z2=
LUT2
H2

【図１８】

LUT1(ビット15～8)

INDEX	OUT (R,G,B,αのいずれか)
0x00 (00000000)	0x00 (00000000)
⋮	⋮
0x0F (00001111)	0x00 (00000000)
0x10 (00010000)	0x01 (00000001)
⋮	⋮
0x1F (00011111)	0x01 (00000001)
0x20 (00100000)	0x02 (00000010)
⋮	⋮
0x2F (00101111)	0x02 (00000010)
0x30 (00110000)	0x03 (00000011)
⋮	⋮
0xE0 (11100000)	0x0E (00001110)
⋮	⋮
0xEF (11101111)	0x0E (00001110)
0xF0 (11110000)	0x0F (00001111)
0xF1 (11110001)	0x0F (00001111)
0xF2 (11110010)	0x0F (00001111)
⋮	⋮
0xFF (11111111)	0x0F (00001111)

【図１９】

LUT2(ビット23～16)

INDEX	OUT (R,G,B,αのいずれか)
0x00 (00000000)	0x00 (00000000)
0x01 (00000001)	0x10 (00010000)
0x02 (00000010)	0x20 (00100000)
0x03 (00000011)	0x30 (00110000)
0x04 (00000100)	0x40 (01000000)
⋮	⋮
0x0E (00001110)	0xE0 (11100000)
0x0F (00001111)	0xF0 (11110000)
0x10 (00010000)	0xF0 (11110000)
0x11 (00010001)	0xF0 (11110000)
⋮	⋮
0xFF (11111111)	0xF0 (11110000)

Q1

クランプ

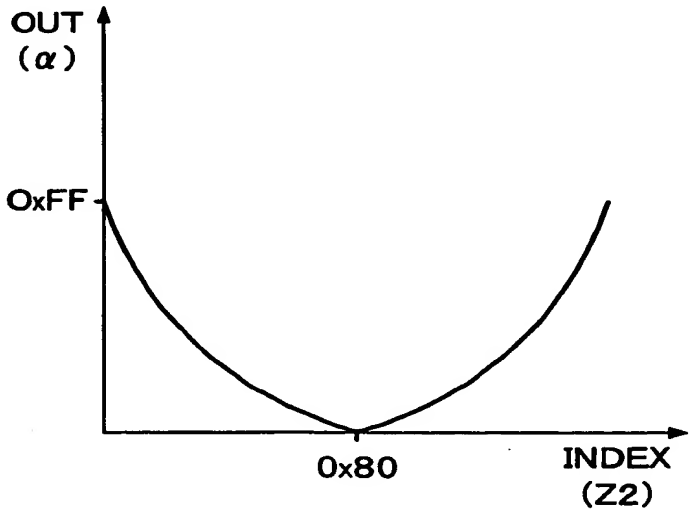
【図２０】

(A)

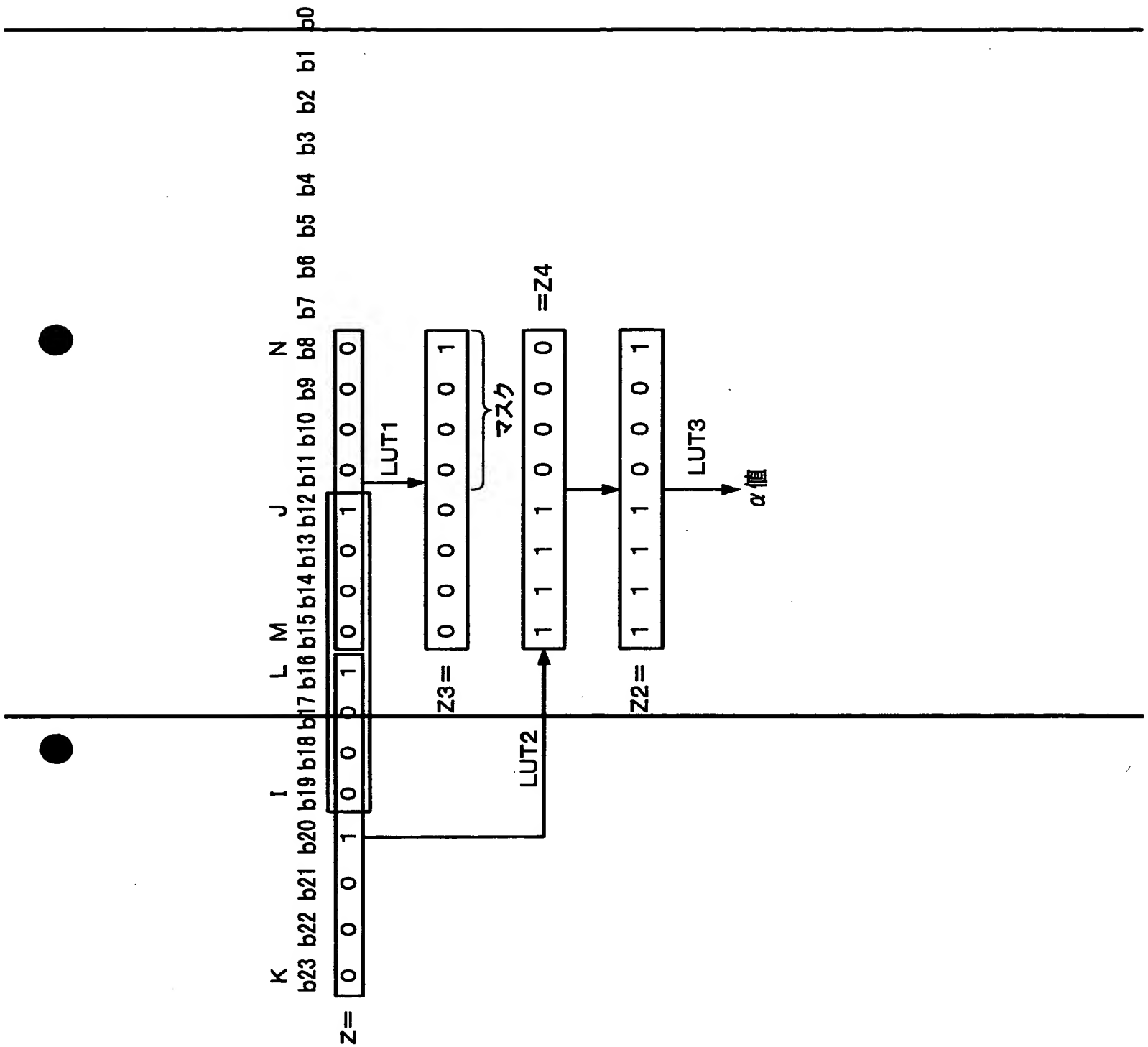
LUT3

INDEX	OUT(α)
0x00 (00000000)	0xFF (11111111)
0x01 (00000001)	0xFE (11111110)
0x02 (00000010)	0xFB (11111011)
⋮	⋮
0x7F (01111111)	0x00 (00000000)
0x80 (10000000)	0x00 (00000000)
0x81 (10000001)	0x00 (00000000)
⋮	⋮
0xFE (11111110)	0xFE (11111110)
0xFF (11111111)	0xFF (11111111)

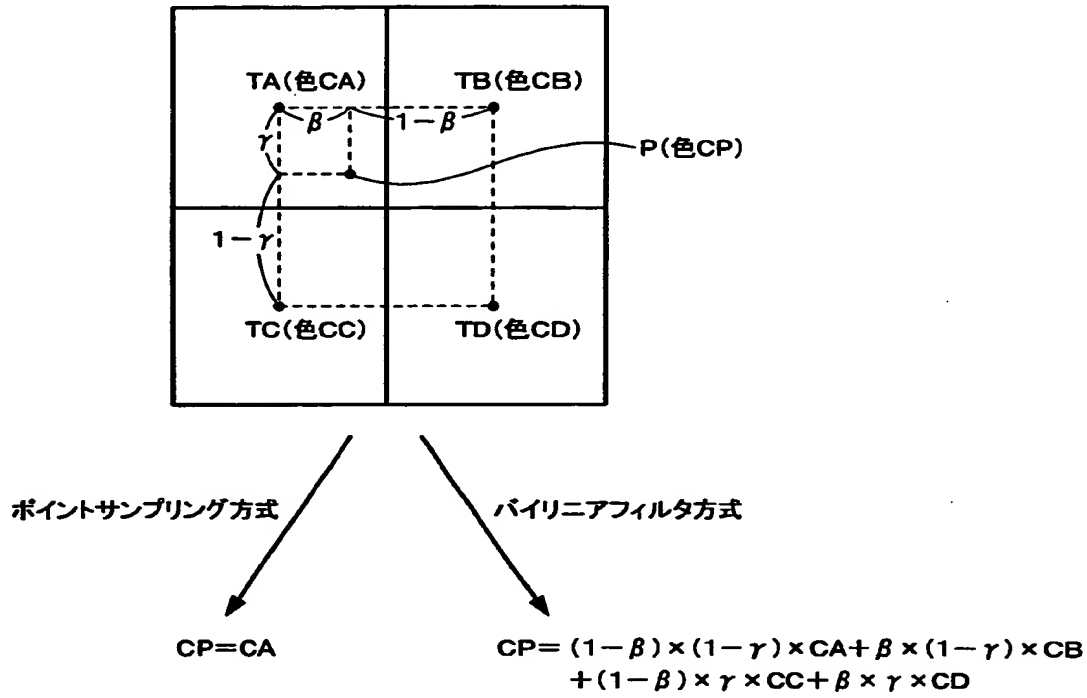
(B)



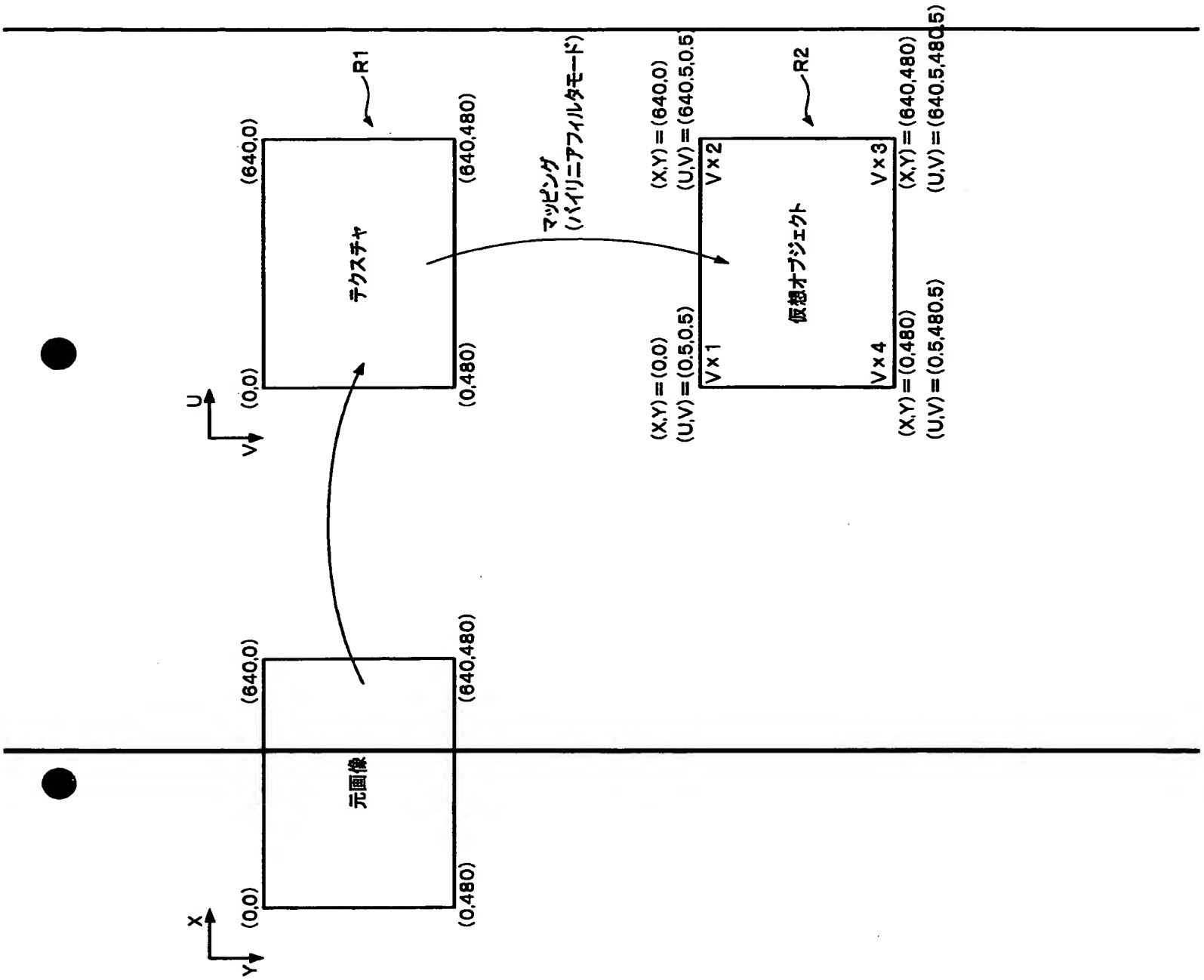
【図２１】


b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 b10 b11 b12 b13 b14 b15 b16 b17 b18 b19 b20 b21 b22 b23
K
I
J
L M
N
Z=
LUT1
Z3=
マスク
=Z4
LUT2
Z2=
LUT3
α値

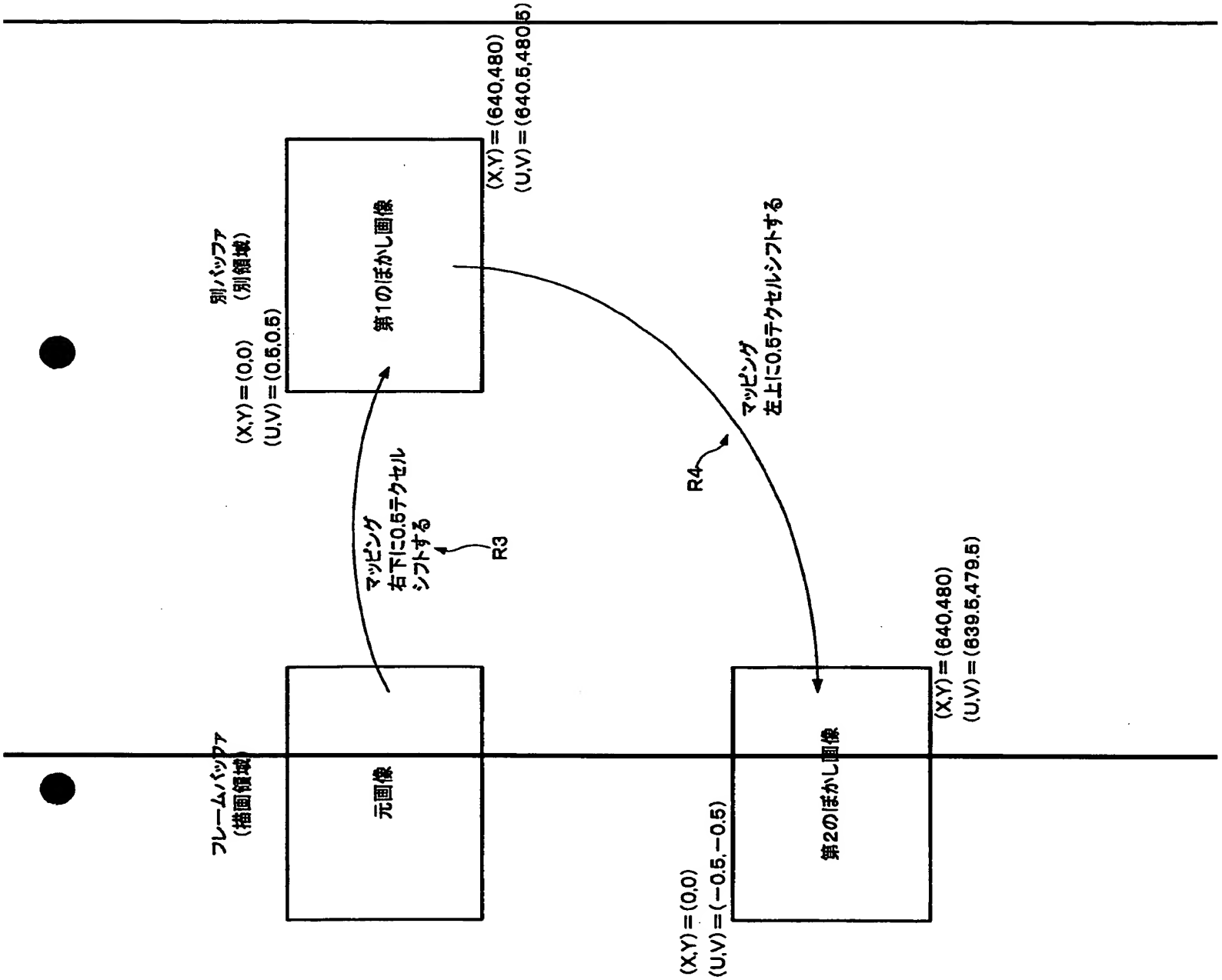
【図２２】


TA(色CA)
TB(色CB)
β
1－β
γ
P(色CP)
1－γ
TC(色CC)
TD(色CD)
ポイントサンプリング方式
バイリニアフィルタ方式
CP＝CA
CP＝(1－β)×(1－γ)×CA＋β×(1－γ)×CB
＋(1－β)×γ×CC＋β×γ×CD

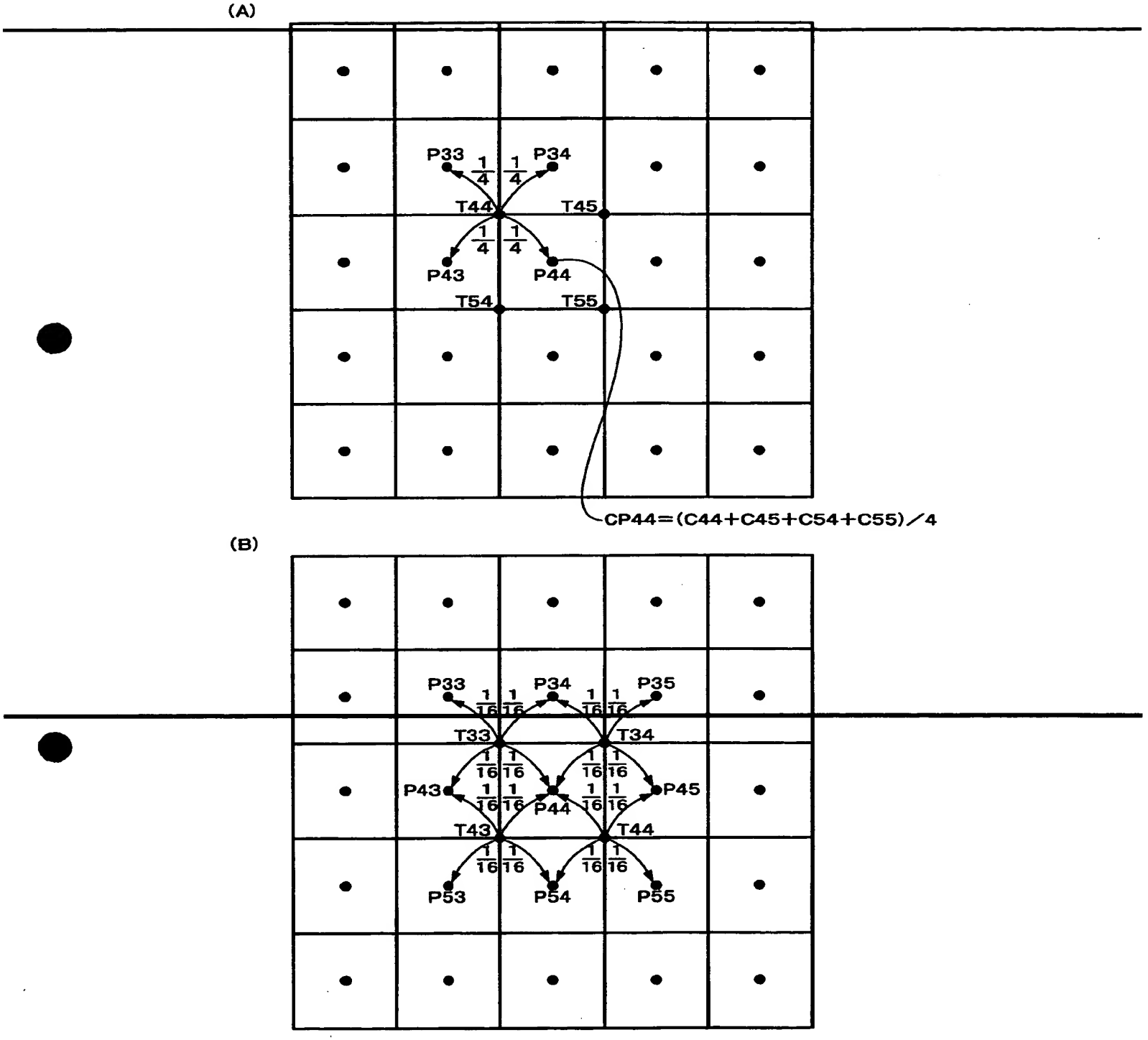
【図２３】


R1
(640,0)
(640,480)
テクスチャ
(0,0)
(0,480)
U
V
マッピング
(バイリニアフィルタモード)
(X,Y)=(640,0)
(U,V)=(640.5,0.5)
V×2
R2
V×3
(X,Y)=(640,480)
(U,V)=(640.5,480.5)
仮想オブジェクト
(X,Y)=(0,0)
(U,V)=(0.5,0.5)
V×1
V×4
(X,Y)=(0,480)
(U,V)=(0.5,480.5)
(640,0)
(640,480)
元画像
(0,0)
(0,480)
X
Y

【図２４】


フレームバッファ
(描画領域)
元画像
別バッファ
(別領域)
(X,Y)=(0,0)
(U,V)=(0.5,0.5)
第1のぼかし画像
(X,Y)=(640,480)
(U,V)=(640.5,480.5)
マッピング
右下に0.5テクセル
シフトする
R3
R4
マッピング
左上に0.5テクセルシフトする
(X,Y)=(640,480)
(U,V)=(639.5,479.5)
第2のぼかし画像
(X,Y)=(0,0)
(U,V)=(−0.5,−0.5)

【図２５】


(A)
P33
P34
T44
T45
P43
P44
T54
T55
CP44=(C44+C45+C54+C55)／4
(B)
P33
P34
P35
T33
T34
P43
P44
P45
T43
T44
P53
P54
P55

【図２６】

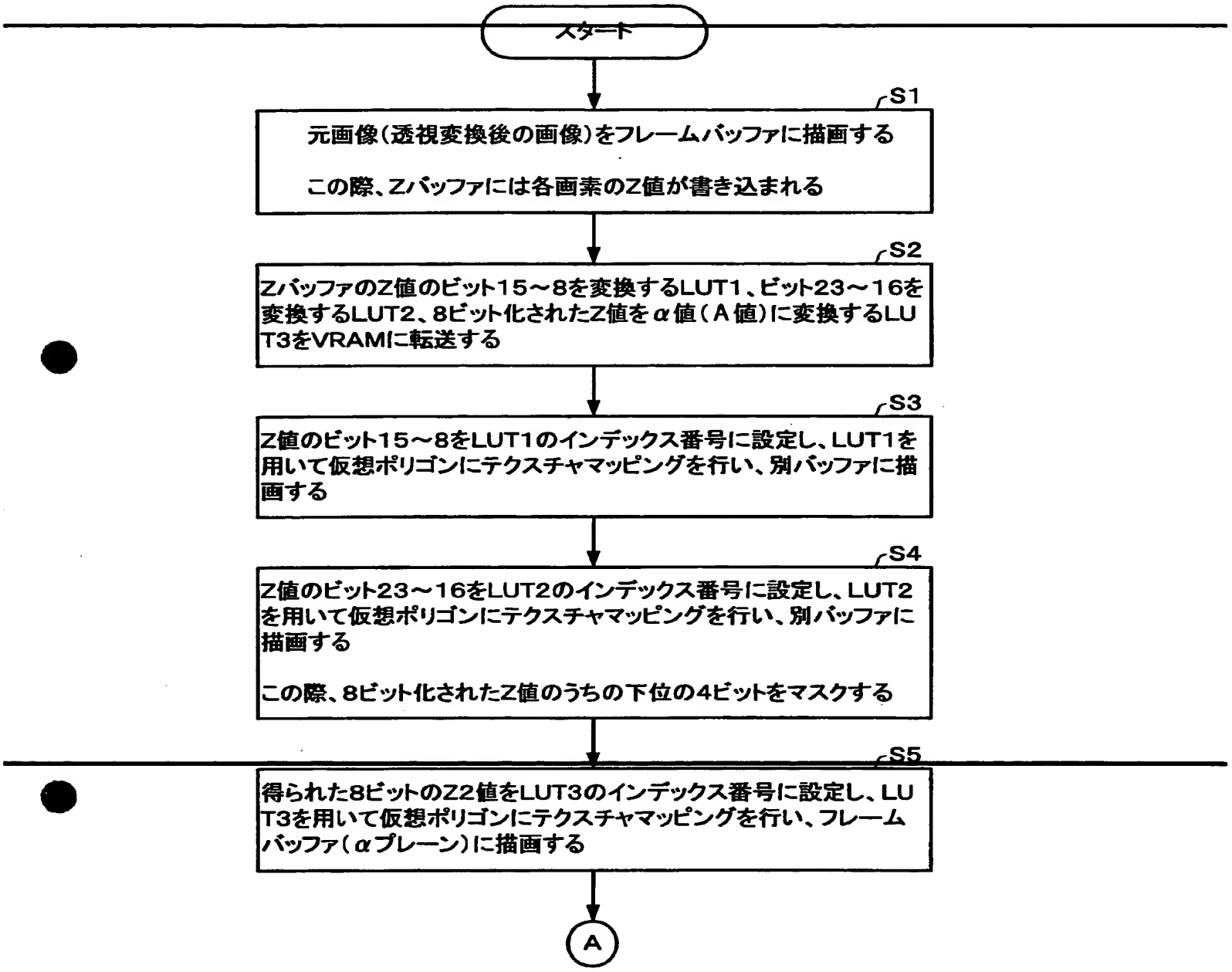
(A)

	$\frac{1}{16}$	$\frac{2}{16}$	$\frac{1}{16}$	
	$\frac{2}{16}$	$\frac{4}{16}$	$\frac{2}{16}$	
	$\frac{1}{16}$	$\frac{2}{16}$	$\frac{1}{16}$	

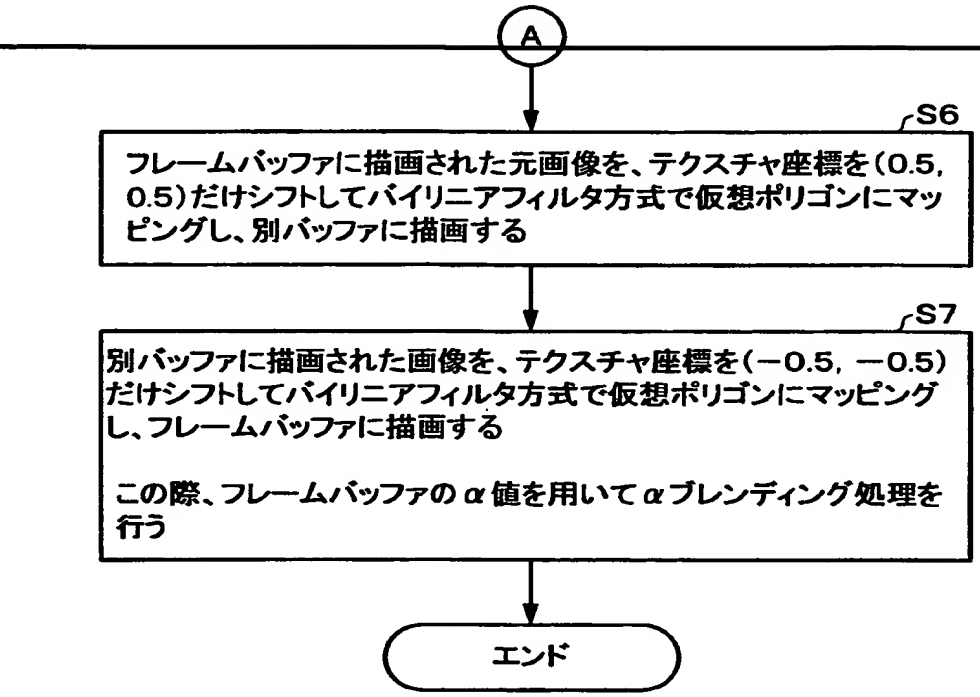
(B)

$\frac{1}{256}$	$\frac{4}{256}$	$\frac{6}{256}$	$\frac{4}{256}$	$\frac{1}{256}$
$\frac{4}{256}$	$\frac{16}{256}$	$\frac{24}{256}$	$\frac{16}{256}$	$\frac{4}{256}$
$\frac{6}{256}$	$\frac{24}{256}$	$\frac{36}{256}$	$\frac{24}{256}$	$\frac{6}{256}$
$\frac{4}{256}$	$\frac{16}{256}$	$\frac{24}{256}$	$\frac{16}{256}$	$\frac{4}{256}$
$\frac{1}{256}$	$\frac{4}{256}$	$\frac{6}{256}$	$\frac{4}{256}$	$\frac{1}{256}$

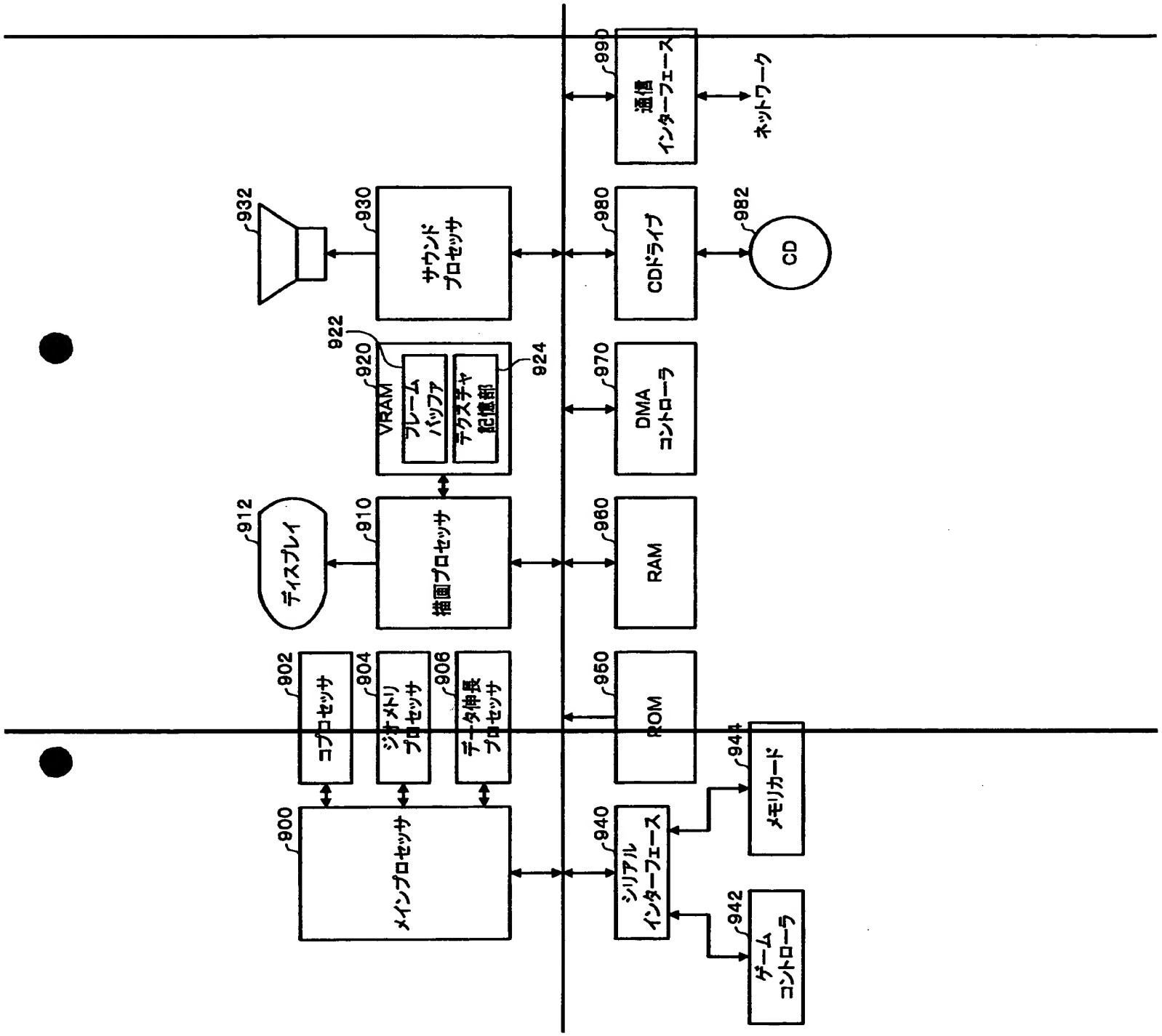
【図２７】


スタート
S1
元画像（透視変換後の画像）をフレームバッファに描画する
この際、Zバッファには各画素のZ値が書き込まれる
S2
ZバッファのZ値のビット15～8を変換するLUT1、ビット23～16を変換するLUT2、8ビット化されたZ値をα値（A値）に変換するLUT3をVRAMに転送する
S3
Z値のビット15～8をLUT1のインデックス番号に設定し、LUT1を用いて仮想ポリゴンにテクスチャマッピングを行い、別バッファに描画する
S4
Z値のビット23～16をLUT2のインデックス番号に設定し、LUT2を用いて仮想ポリゴンにテクスチャマッピングを行い、別バッファに描画する
この際、8ビット化されたZ値のうちの下位の4ビットをマスクする
S5
得られた8ビットのZ2値をLUT3のインデックス番号に設定し、LUT3を用いて仮想ポリゴンにテクスチャマッピングを行い、フレームバッファ（αプレーン）に描画する
A

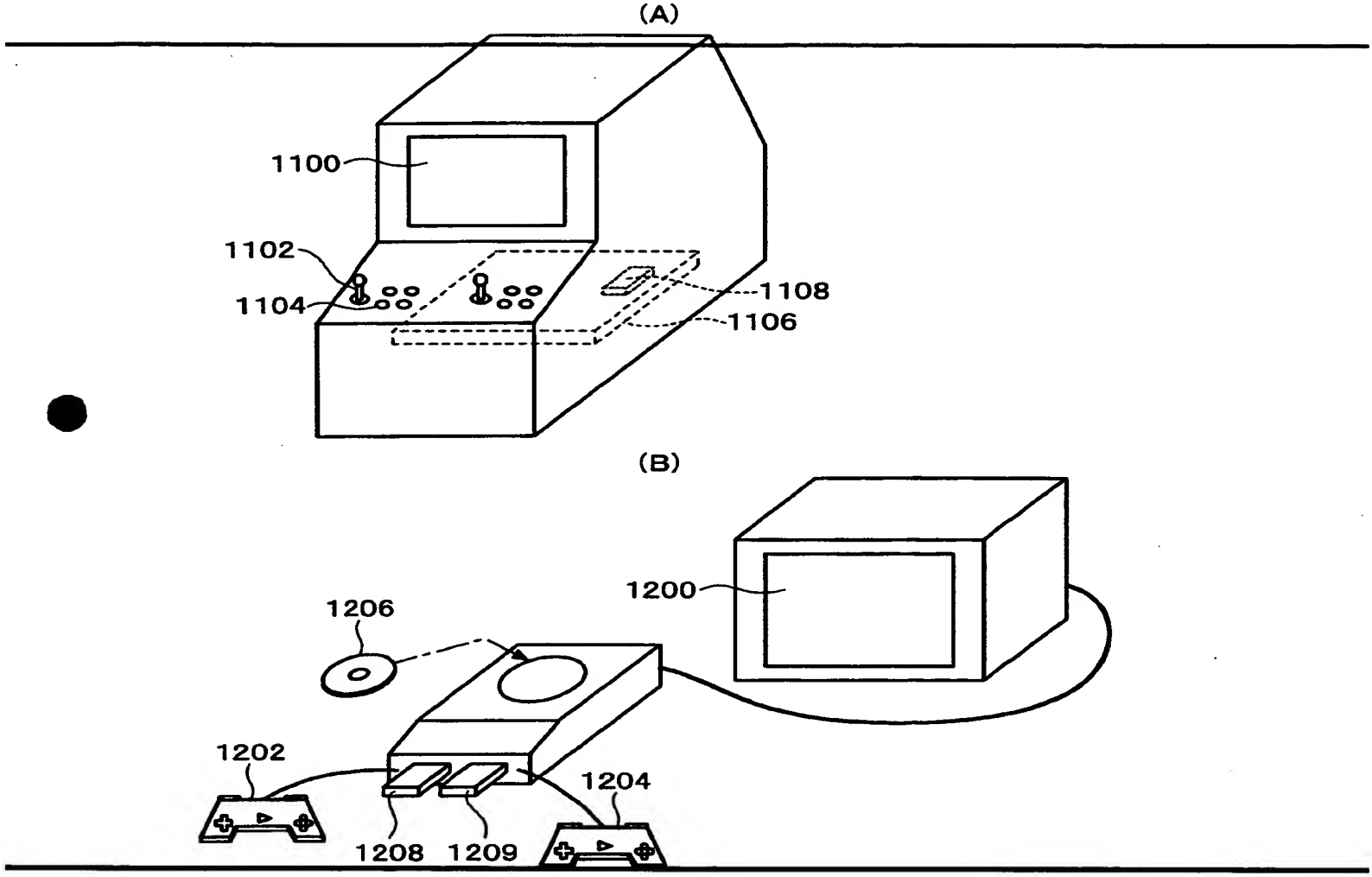
【図２８】


A
S6
フレームバッファに描画された元画像を、テクスチャ座標を（0.5，0.5）だけシフトしてバイリニアフィルタ方式で仮想ポリゴンにマッピングし、別バッファに描画する
S7
別バッファに描画された画像を、テクスチャ座標を（－0.5，－0.5）だけシフトしてバイリニアフィルタ方式で仮想ポリゴンにマッピングし、フレームバッファに描画する
この際、フレームバッファのα値を用いてαブレンディング処理を行う
エンド

【図２９】


メインプロセッサ
800
コプロセッサ
902
ジオメトリプロセッサ
904
データ伸長プロセッサ
906
描画プロセッサ
910
ディスプレイ
912
VRAM
920
フレームバッファ
テクスチャ記憶部
922
924
サウンドプロセッサ
930
932
通信インターフェース
990
ネットワーク
CDドライブ
980
CD
982
DMAコントローラ
970
RAM
960
ROM
950
シリアルインターフェース
940
メモリカード
944
ゲームコントローラ
942

【図３０】


(A)
1100
1102
1104
1106
1108
(B)
1200
1206
1202
1204
1208
1209
(C)
ホスト装置
1300
情報記憶媒体
1306
1302
端末
端末
端末
端末
1304－1
1304－2
1304－3
1304－n

【書類名】　　要約書

【要約】

【課題】　現実世界の視界画像のようにフォーカシングされた画像を、少ない処理負担で生成できるゲームシステム及び情報記憶媒体を提供すること。

【解決手段】　元画像の各画素のＺ値を、インデックスカラー・テクスチャマッピング用のルックアップテーブルＬＵＴのインデックス番号に設定して仮想オブジェクトに対するテクスチャマッピングを行い、元画像の各画素のＺ値に応じた値にα値を設定し、元画像とそのぼかし画像を合成する。Ｚ値を、その最上位ビットよりも下位のビットＩ～Ｊにより構成されるＺ２値に変換し、Ｚ２値をＬＵＴのインデックス番号に設定する。Ｚ値のビットＩ～Ｊ以外のビットの値に応じてＺ２値を所与の値にクランプする。Ｚ値のビットＭ～Ｎ、ビットＫ～Ｌ（Ｋ≧Ｉ≧Ｌ＞Ｍ≧Ｊ≧Ｎ）をＬＵＴ１、ＬＵＴ２のインデックス番号に設定してテクスチャマッピングを行い、Ｚ値をＺ３値、Ｚ４値に変換し、Ｚ３値、Ｚ４値に基づきＺ２値を求める。

【選択図】　　図１４

出願人履歴情報

識別番号　〔０００１３４８５５〕

１．変更年月日　１９９０年　８月２３日
［変更理由］　新規登録
住　所　東京都大田区多摩川２丁目８番５号
氏　名　株式会社ナムコ